ANOMALY DETECTION USING MACHINE LEARNING

By

Kandagatla Uttej Kumar – 19BAI1015

Potti Sai Pavan Guru Jayanth – 19BAI1045

A project report submitted to

Dr. Anusha K

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

in partial fulfillment of the requirements for the course of

CSE3501 – Information Security Analysis and Audit

B. TECH Computer Science Engineering with Specialization in Artificial Intelligence and Machine Learning

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

August - December 2021

DECLARATION BY THE CANDIDATE

I hereby declare that the report titled "Anomaly Detection Using Machine Learning**"** submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. Anusha K, Associate Professor, SCOPE, Vellore Institute of Technology, Chennai.**

Signature of the Candidate

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Anusha K,** School of Computer Science and Engineering for her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. Ganesan R, Dean,** School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the school towards our project and for his unstinting support.

We express our thanks to our **Head of the Department** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

BONAFIDE CERTIFICATE

Certified that this project report entitled "**ANOMALY DETECTION USING MACHINE LEARNING**" is a bona-fide work of **Kandagatla Uttej Kumar (19BAI015)**, **Potti Sai Pavan Guru Jayanth (19BAI1045)** carried out the **"J"-Project** work under my supervision and guidance for **Subject CSE3501 – Information Security Analysis and Audit**.

Dr. Anusha K
SCOPE

INDEX

ABSTRACT

Anomaly detection has become one of the essential tools in every field as it identifies the abnormal behaviours across data logs that are not identified when using traditional security methods. In today's world the design principles of the internet do not have a check point of how many requests the HTTP receives. As a result, there are more chances that it undergoes an attack, one abnormal situation is Distributed Denial of Service attack.

Distributed Denial of Service (DDoS) attack is something that is highlighted in today's cyberworld. It specifically aims to bring the server down, by interrupting the normal server or network operation. The main aim of this report is to detect the chances of DDoS attack in a server by using number of counts to detect anomalous behaviour of Ip addresses in the data logs of the User-app.

This report will be ending in a way by justifying the name of the project "Anomaly Detection Using Machine Learning", successfully detecting anomalies in the data logs that are in hand using unsupervised learning method and Ensemble Techniques. By choosing 3 algorithms, applying them to the data and after identifying anomalies in the data, the data becomes Binary Classification then I tried to implement Ensemble techniques to the data then comparing all the results to get an idea of which algorithm is best and accurate for anomaly detection.

Keywords: Anomaly Detection, Unsupervised Algorithms, DDoS attack, K-means, Isolated Forest, Clustering, Possible attack vectors, Ensemble Techniques

INTRODUCTION

Anomalies are considered to be the abnormal behaviour or something that deviates from the standard, normal or expected action. For example, if a person is changing their password once in a month from long time and suddenly that accounts password changes two to three times a week, this abnormal behaviour is an anomaly and finding this sort of behaviour in early stages is called Anomaly Detection (AD).

Anomaly detection techniques constructs an overall view of expected behaviour that would be normal using data logs that are collected from long period of normal system activity. These data logs are collected in an environment that is secured, then it is analysed and Dept in a proper way to ensure that the anomaly detection is possible. When the anomaly detection is applied, the system sometimes detects false alarms due to the difficulty in obtaining the normal behaviour of the system.

Sometimes the system May detect or generate excessive number of false alarms and report as anomalous events, which are not necessarily abnormal activities. An example is the situation in fig [1]. There, two areas N1 and N2 (N for normal) are formed of points of a data amount. These data points are composed of values of two different features x and y. In addition, there is the expectation that newly added points in these two areas accumulate. Data points of these two quantities are thus considered normal Basics. Now there are two points O1, O2 and the set of points O3, all far away from both normal amounts, these may be considered as anomalies.

In general, therefore, the anomaly can be described as something that depends on the expectation and deviates to a certain extent. One of the advantages is, the configuration of the system can be unattended as it automatically learns the behaviour of large number of subjects and can run itself. The drawback is that there is a chance of accepting the abnormal behaviours as normal for a particular user. When it slowly gets adapted to this kind of behaviour, it might consider the attack as normal behaviour.

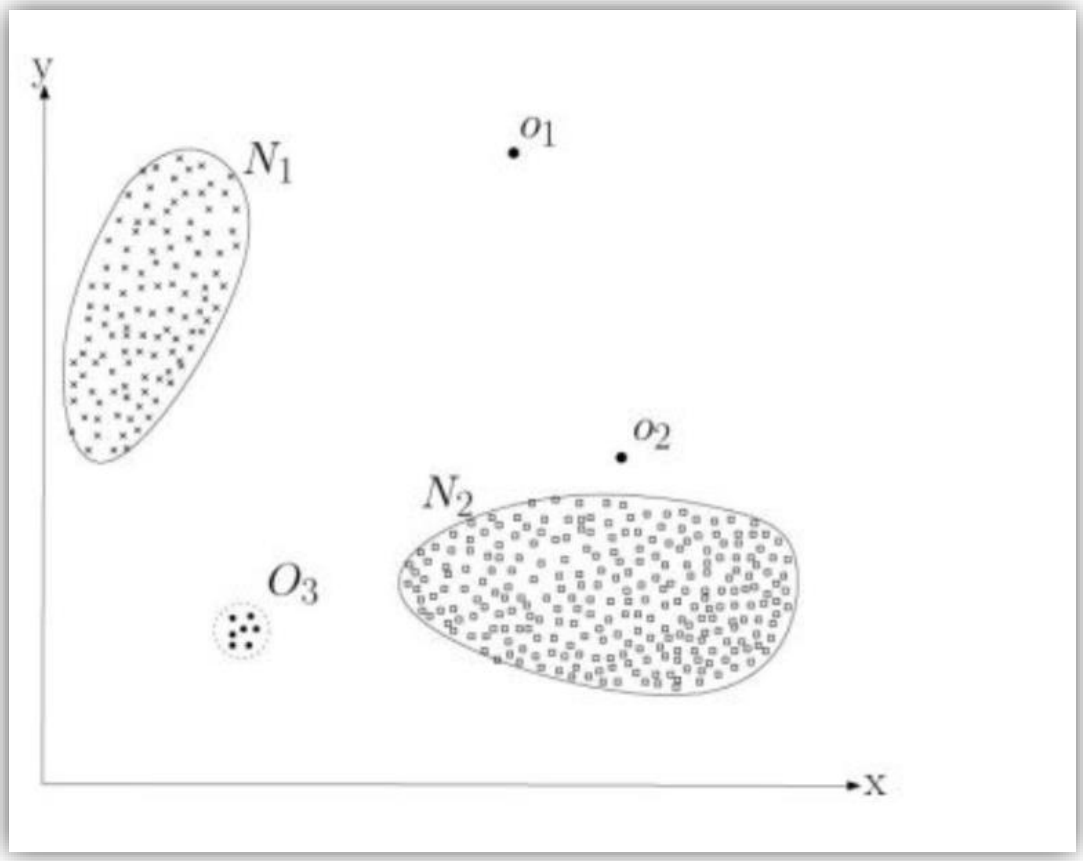

fig [1]

RELATED WORK

Log-based Anomaly Detection

There are many studies done to detect anomalies based on logs. Current approaches are mainly divided into three categories: supervised learning methods, unsupervised learning methods, and deep learning methods. Many supervised learning methods are used for log-based anomaly detection. For example, [4] trained an SVM classifier to detect failures using event logs. [3] used a decision tree model to detect anomalies in application performance. [5] proposed a regression-based anomaly detection method using log data from cloud systems. [2] summarizes several classic supervised classification models used for log-based anomaly detection.

In addition to supervised learning approaches, many unsupervised learning approaches have been proposed. For example, Lou et al. [6] proposed invariance analysis (IM) to find the invariance (linear relationship) between log events in a vector of log event counts. A log sequence that violates an invariant relationship is considered an anomalous sample. Xu et al. [7] used principal component analysis to construct the stationary and anomalous spaces of log event scoring matrices to detect anomalies. Line et al. [8] and He et al. [9] developed a clustering-based method to identify problems in online service systems. The unsupervised learning approach has the advantage that no manual signals are required on the training set.

The recent rise of deep learning technology has provided new log-based anomaly detection solutions. [10] used LSTM to predict log sequence anomalies based on log keys. [11] LSTMs were also used to predict the next log event and then compared it to the currently valid data to detect anomalies. [12] They trained stackedLSTM to simulate normal and anomalous event log samples.

DATASET

The dataset is a logs data from a remote server generated for 1 month. This dataset is created, post cleaning and picking only relevant events on which we wish to identify anomalies by Kibana.

This data has three columns:

@timestamp: The timestamp that describes when the login happens.

id: The id field gives the unique identity of the users.

ip_address: These Ip addresses are of user's Operating System.

Total data that is used to detect anomalies is 721547 rows x 3 columns (timestamp, user_id and ip_address)

	@timestamp	_id	ip_address
0	July 8th 2019, 14:43:03.000	XswJ0msBoTGddM7vxMDB	10.1.1.285
1	July 8th 2019, 14:43:01.000	dKQJ0msB7mP0GwVzvJjz	10.1.2.389
2	July 8th 2019, 14:42:59.000	CcwJ0msBoTGddM7vtb8y	10.1.1.415
3	July 8th 2019, 14:42:57.000	bKQJ0msB7mP0GwVzrZdT	10.1.1.79
4	July 8th 2019, 14:42:55.000	L6QJ0msB7mP0GwVzpZel	10.1.1.60

fig [2]

ARCHITECTURE

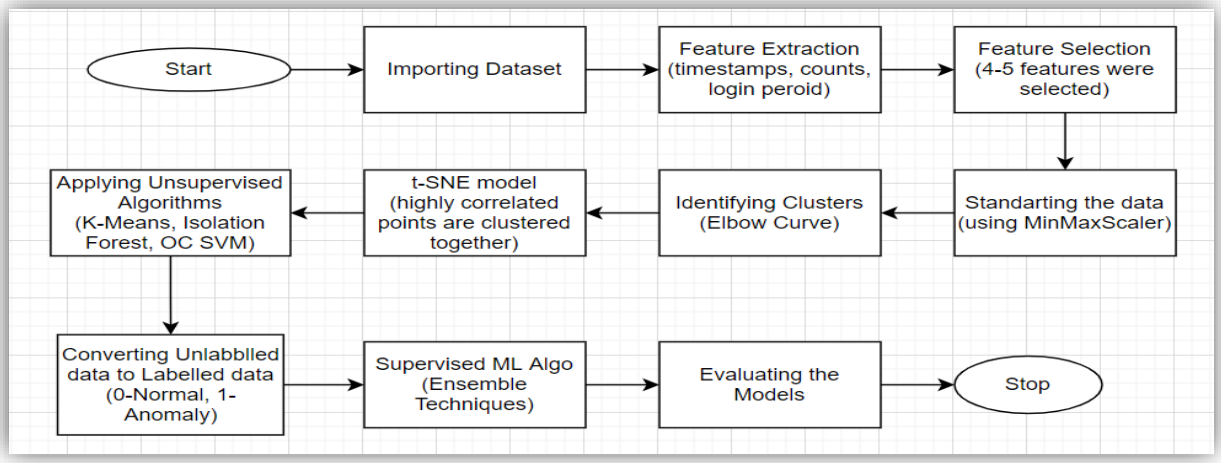

fig [3]

ALGORITHM IMPLEMENTATION

1. K-means Implementation

Cluster analysis is recognized as an important technique for classifying data, finding clusters of a dataset based on similarities in the same cluster and

dissimilarities between different clusters. Putting each point of the dataset to exactly one cluster is the basic of the conventional clustering method whereas clustering algorithm actually separates unlabelled set of data into different groups according to the similarity. Clustering is basically considered as classification of similar objects, it is precisely partitioning of datasets into clusters so that data in each cluster shares some common trait. Now the Kmeans which is considered to be the main clustering algorithm is implemented based on the clusters.

Fig [4] is a pairwise plot of the features. This graph is used to understand the relationship between the variables. While most relationships are looking like clusters, the td_max vs is_weekend_ratio is a giant blob. As t-SNE is good at understanding non linear relationships, we use t-SNE to explain this graphs. The main purpose of t-SNE was to visualize the blobs.

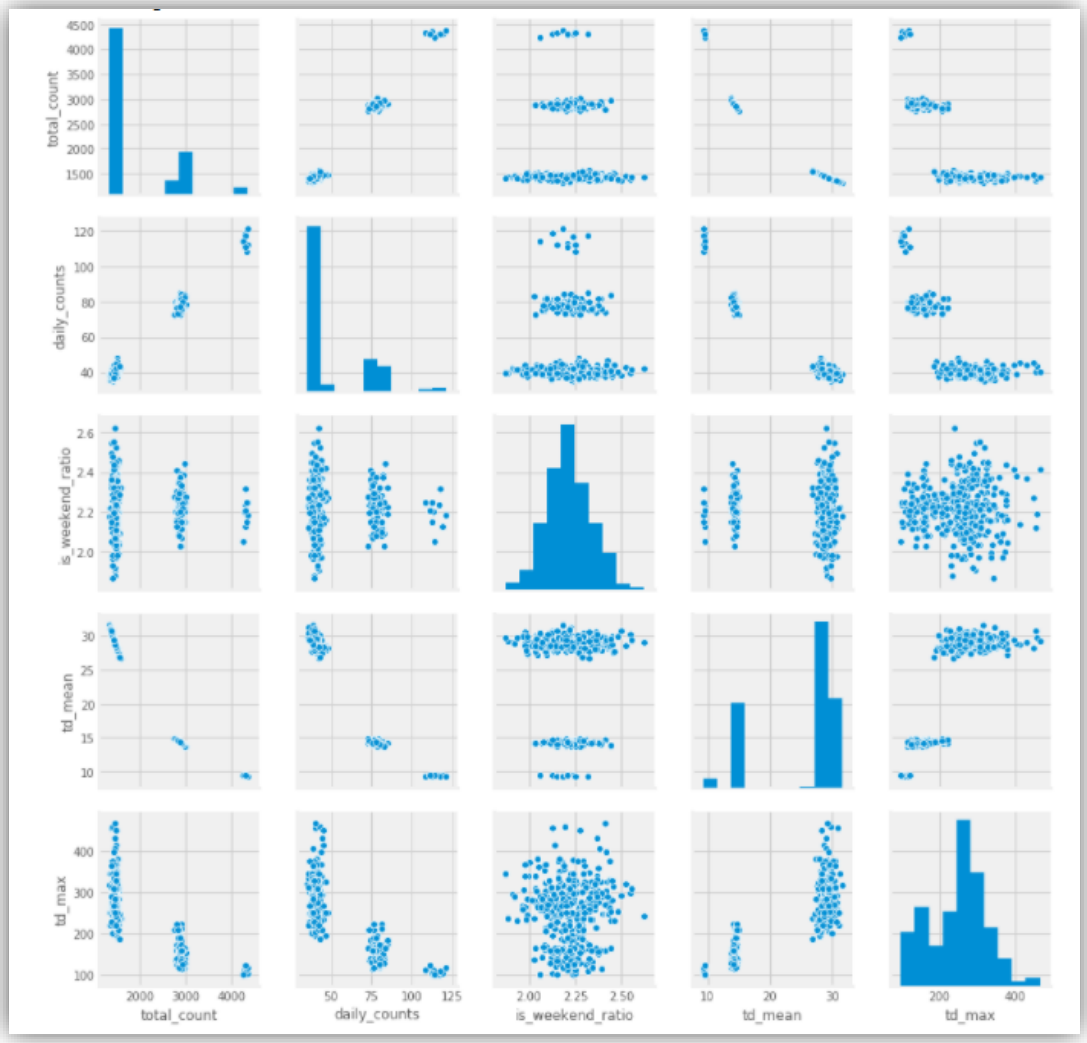

fig [4]

The Elbow curve(fig [5]) is something like an interpretation and validation of consistency within cluster analysis, this is mainly used to find the appropriate number of clusters in a dataset. From the below elbow curve, we see that the graph levels off after 6 clusters, implying that addition of more clusters do not explain much more of the variance in our relevant variable. The number of clusters chosen is therefore 6.

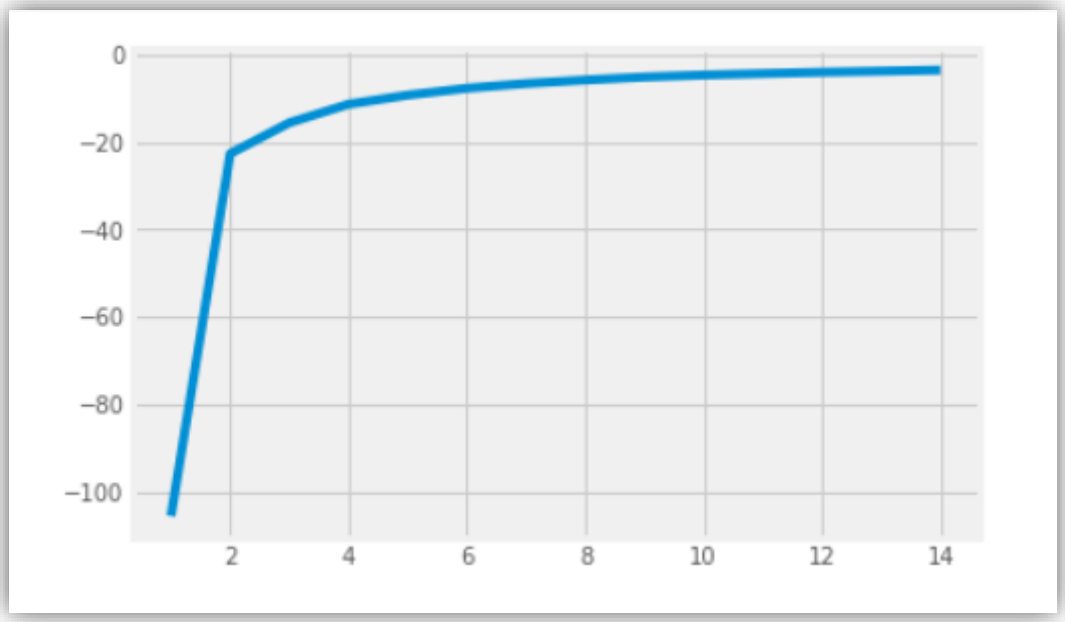

fig [5]

1.1. Clustering model with t-SNE

t-SNE- plots(fig [6]) are mainly used for creating a graph that has a high dimensional data logs and reduces into low dimensional data points but has a lot of the original information in it. This is just to reduce the same feature points and to convert the correlations in the data logs into a 2-Dgraph as shown below in figure. Data points that are highly correlated are clustered together. The axis in the 2-D graph are not named by our choice, but it is done by t-SNE according to their importance. That is the reason to leave the axis named as t-SNE-2d-one and t-SNE--2d-two. The differences along the t-sne-2d-one are more important compared to t-sne-2d-two.

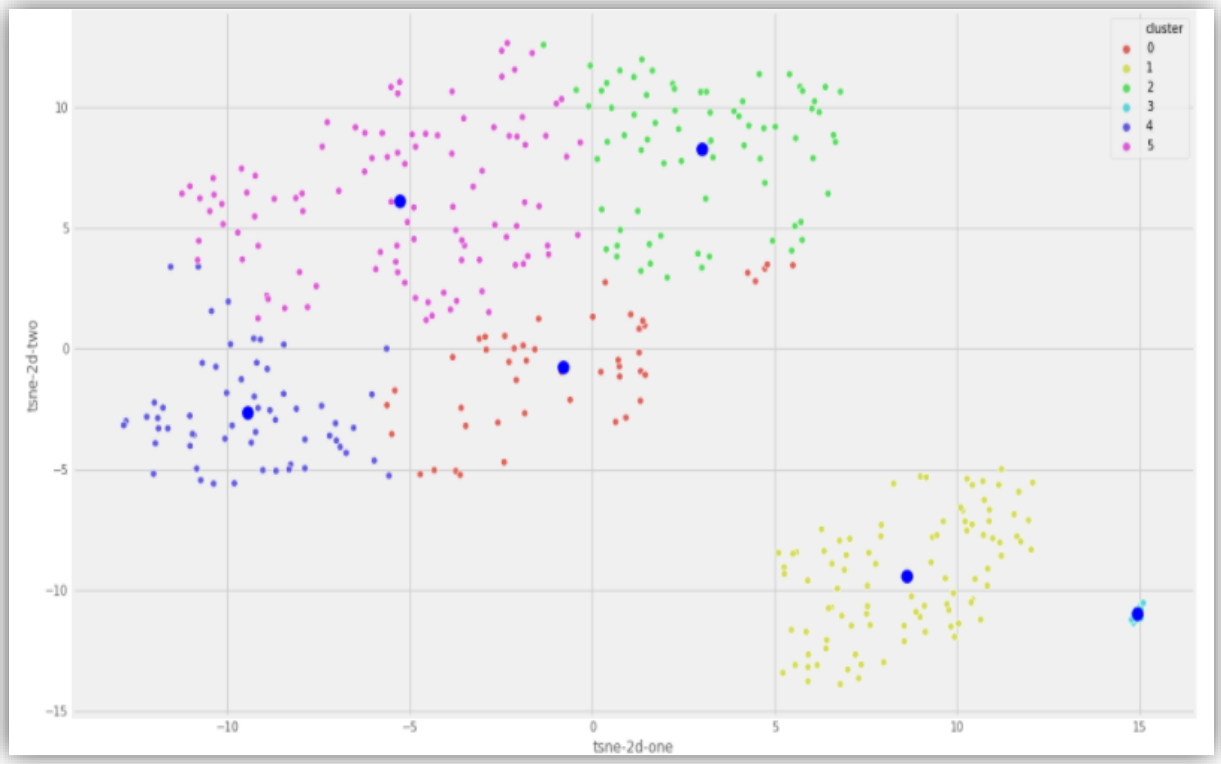

fig [6]

1.2. Bar graph of cluster_model

In this sub-session the sum of square distance' (ssd)is calculated and plotted in the graph. Each ip_address's distance as per timestamp is calculated and the values are added to the graph. The distance that is near to the coordinate 0 are considered as the normal behaviour and here far from it are anomalous.

cutoff = 6(from fig [7])

The cutoff ratio is considered by knowing the data and depends on how sensitive the algorithm must behave in detecting anomalies. Here we are considering the cutoff point at 6 and applying the algorithm. So considering SSd is greater 6 are anomalies gives better result for Kmeans.

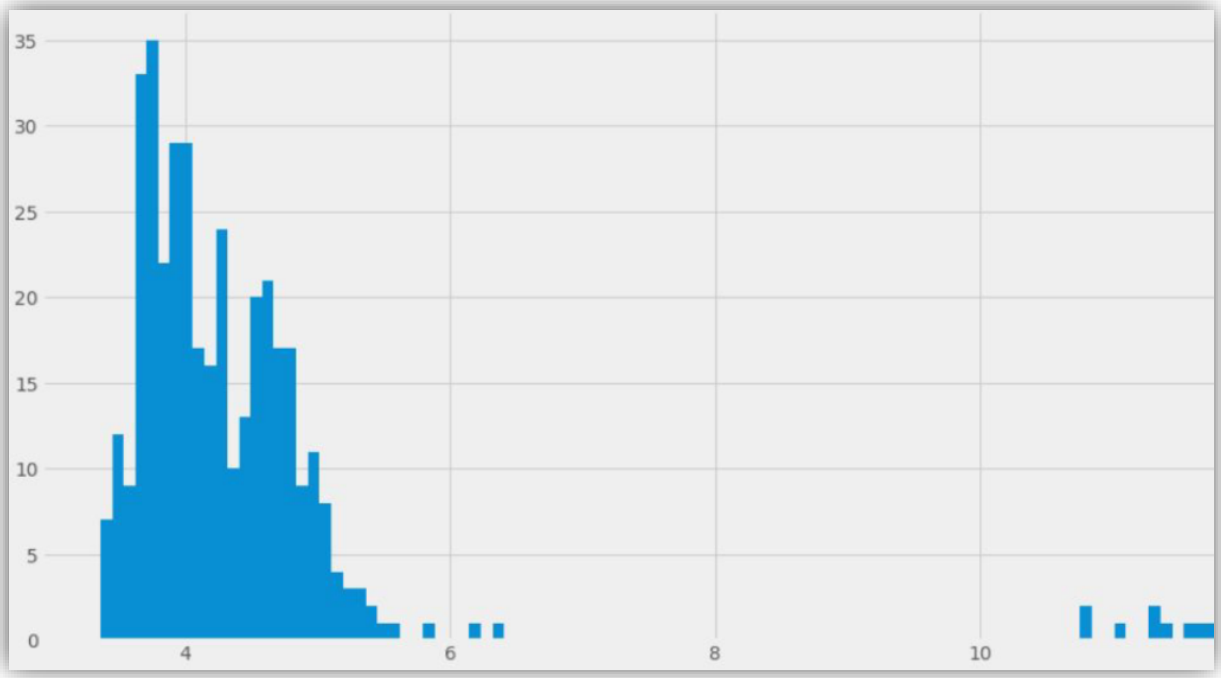

fig [7]

1.3. K-means clustering algorithm

The underlying assumption in the clustering-based anomaly detection is that if we cluster the data, normal data will belong to clusters while anomalies belong to small clusters. We use the following steps to find and visualize anomalies.

Chose the number of clusters then the initialization of the centroid, each data point is assigned to a cluster with closest centroid, Means of each cluster is calculated to be its new centroid after recomputing the centroid, Sum of square distance is calculated, considering the cutoff point at 6 and applying the algorithm. So, considering SSD that is greater than 6 are anomalies gives better result for Kmeans.

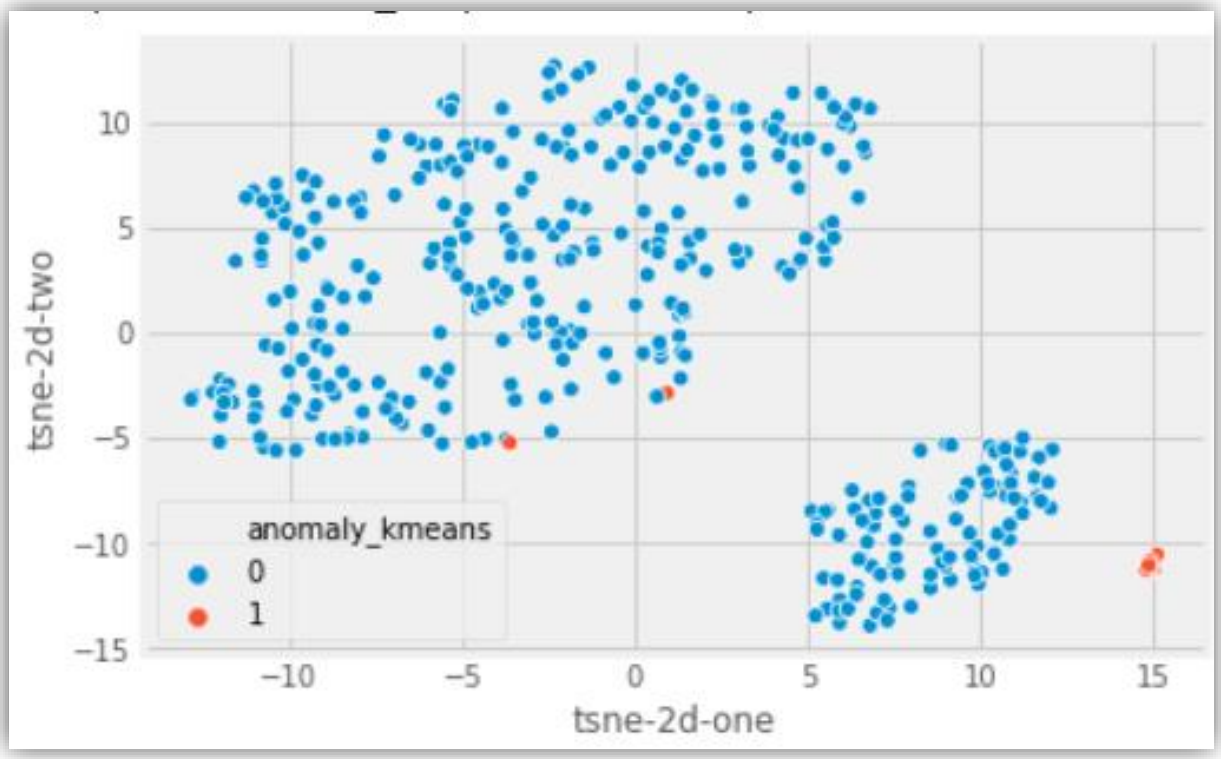

fig [8]

2. Isolation Forest implementation

Isolation forest is implemented in this project using the scikit-learn (sklearn) python libraries. Given the input data set and certain parameters, the sklearn ensemble method Isolated Forest can be used to isolate the outliers, which are in our case the anomalies. The method fit() is used to fit a model to the normalised input data set. The parameter number of estimators is set to 200 and the contamination is set to 0.028, based on initial outlier guess in the data. Further, the predict() method is used to detect if a given data is an outlier or not. When applying an IsolationForest model, we set contamination = outliers_fraction, that is telling the model that the proportion of outliers in the data set is 0.028. fit and predict(data) performs outlier detection on data, and returns 0 for normal, 1 for anomaly. Finally, we visualize anomalies with t-SNE 2d graph. The method returns 0 for normal and 1 in case of outlier or anomaly. For the given data, 375 data points are returned as normal and 11 data points are determined to be anomalies.

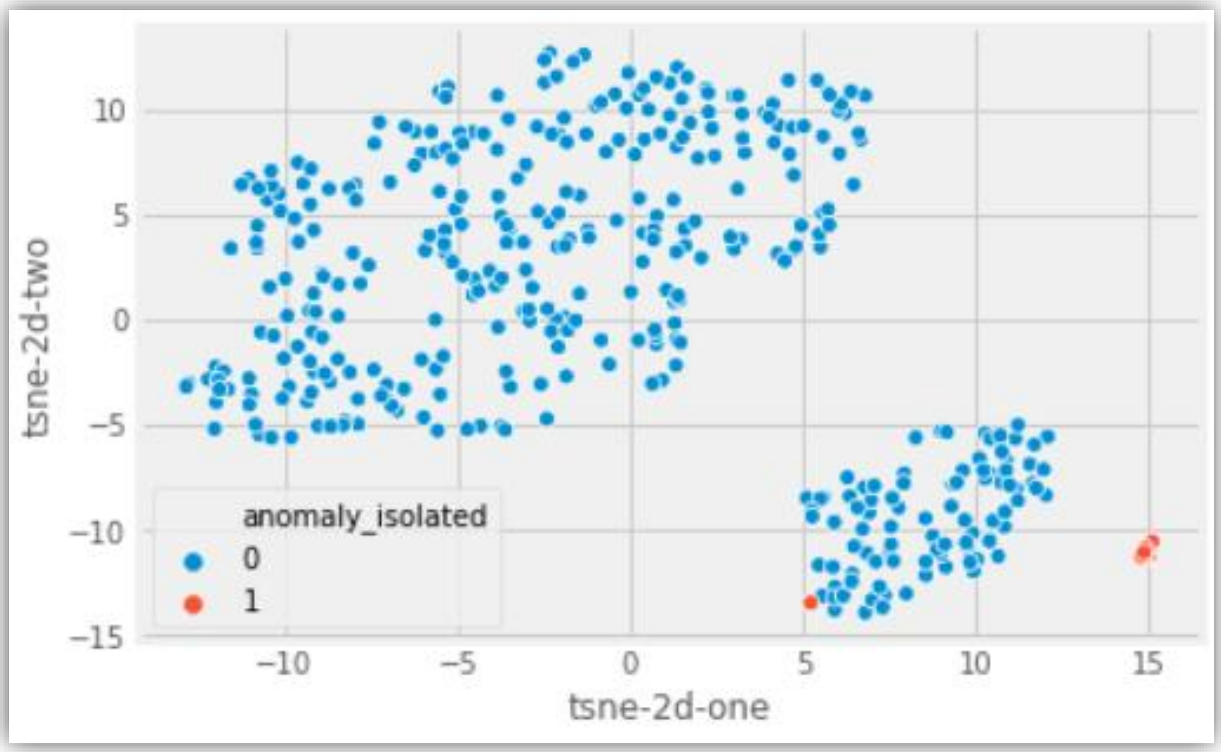

fig [9]

3. One Class SVM

In One Class SVM we select one class to implement this algorithm and detect anomalies. When fitting OneClassSVM model, we set outliers_fraction, which is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors and must be between 0 and 1. Basically this means the proportion of outliers we expect in our data. Specifies the kernel type to be used in the algorithm as in our case its rbf. This will enable OC-SVM to use a nonlinear function to project the hyperspace to higher dimension. Gamma is a parameter of the RBF kernel type and controls the influence of individual training samples - this affects the "smoothness" of the model. Through experimentation, I did not find any significant difference. predict() perform classification on data, and because our model is an one-class model, +1 or 0 is returned, and 1 is anomaly. 0 is normal.

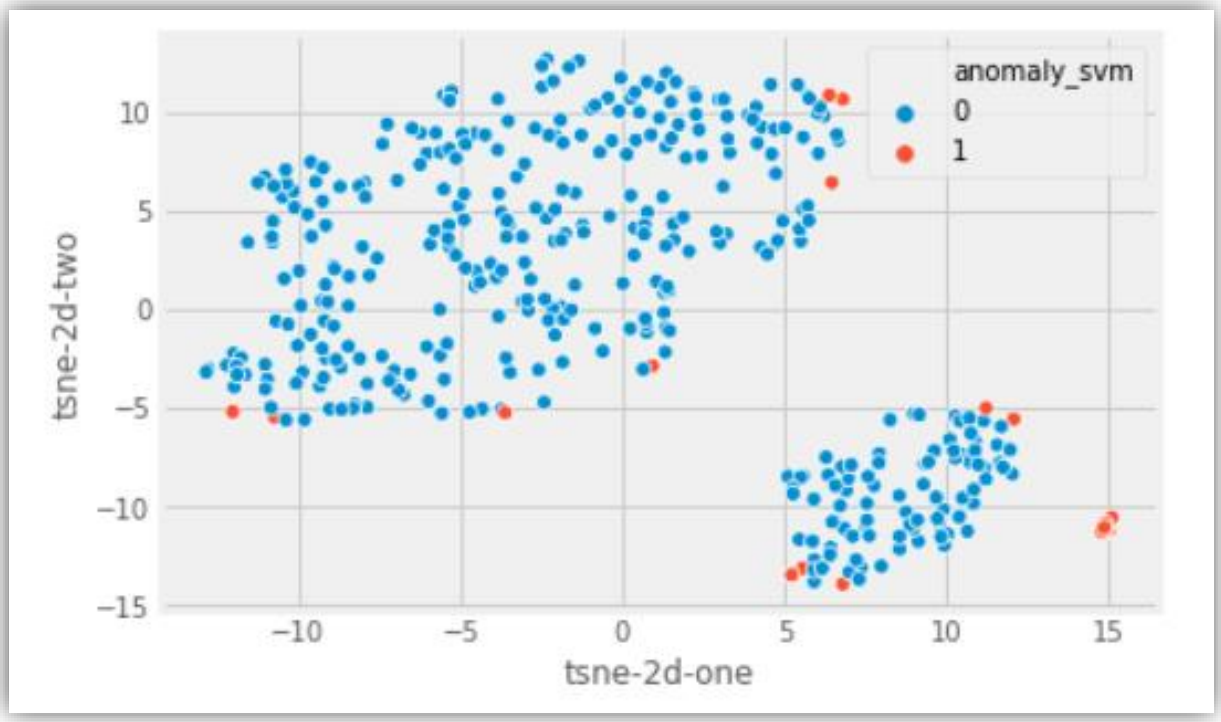

fig [10]

4. Ensemble models for Classification

Stacking mainly differ from bagging and boosting on two points. First stacking often considers heterogeneous weak learners (different learning algorithms are combined) whereas bagging and boosting consider mainly homogeneous weak learners. Second, stacking learns to combine the base models using a meta-model whereas bagging and boosting combine weak learners following deterministic algorithms. So, in our project we want to fit a stacking ensemble composed of L weak learners. Then we have to follow the steps thereafter:

Split the training data in two folds, choose L weak learners and fit them to data of the first fold, for each of the L weak learners, make predictions for observations in the second fold, fit the meta-model on the second fold, using predictions made by the weak learners as inputs

Here in our project, we used KNeighborsClassifier, SVC, GaussianNB, RandomForestClassifier as base-models and for meta-model LogisticRegression.

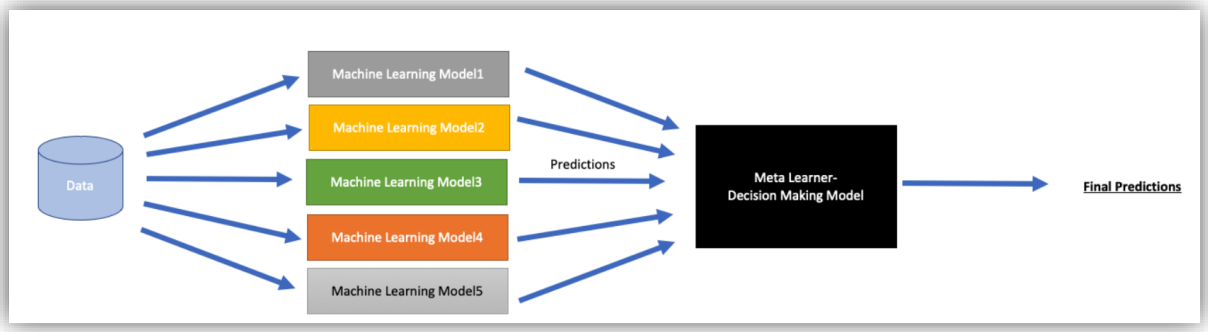

fig [11]

EXPERIMENTALS RESULTS

```
>KNNC 0.689 (0.071)

>SVC 0.966 (0.024)

>RF 0.817 (0.051)

>GNB 0.866 (0.044)

>Stacking 0.967 (0.021)
```

fig [12]

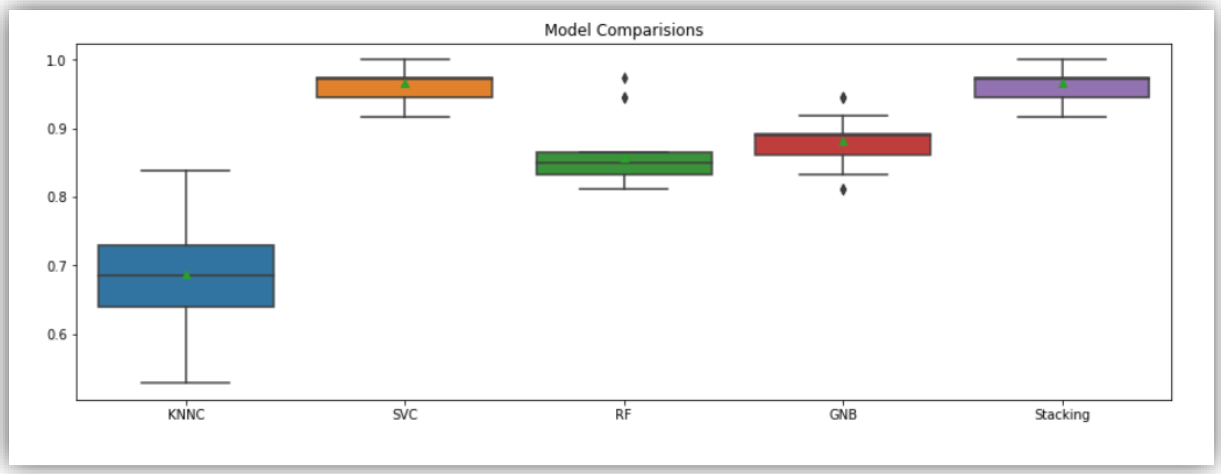

fig [13]

With a stacking model, we were able to get an F1 score of 96.7%, which was greater than the target of 81.1%

CONCLUSION

We successfully detected anomalies in the data logs that are in hand using unsupervised learning method and Ensemble Techniques. By choosing 3 unsupervised clustering algorithms, applying them to the data and after identifying anomalies in the data, the data becomes Binary Classification then we implemented Ensemble techniques Stacking to the data, then we compared all the results to get an idea of which algorithm is best and accurate for anomaly detection.

REFERENCES

[1] Peter Bodik, Moises Goldszmidt, Armando Fox, Dawn B Woodard, and Hans Andersen. 2010. Fingerprinting the datacenter: automated classification of performance crises. In Proceedings of the 5th European conference on Computer systems. ACM, 111–124

[2] Jakub Breier and Jana Branišová. 2017. A dynamic rule creation-based anomaly detection method for identifying security breaches in log records. Wireless Personal Communications 94, 3 (2017), 497–511.

[3] Mike Chen, Alice X Zheng, Jim Lloyd, Michael I Jordan, and Eric Brewer. 2004. Failure diagnosis using decision trees. In null. IEEE, 36–43

[4] Yinglung Liang, Yanyong Zhang, Hui Xiong, and Ramendra Sahoo. 2007. Failure prediction in ibm bluegene/l event logs. In Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on. IEEE, 583–588

[5] Mostafa Farshchi, Jean-Guy Schneider, Ingo Weber, and John Grundy. 2015. Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis. In Software Reliability Engineering (ISSRE), 2015 IEEE 26th International Symposium on. IEEE, 24–34

[6] Jian-Guang Lou, Qiang Fu, Shengqi Yang, Ye Xu, and Jiang Li. 2010. Mining Invariants from Console Logs for System Problem Detection.. In USENIX Annual Technical Conference. 23–25.

[7] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In

Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. ACM, 117–132.

[8] Qingwei Lin, Hongyu Zhang, Jian-Guang Lou, Yu Zhang, and Xuewei Chen. 2016. Log clustering based problem identification for online service systems. In Proceedings of the 38th International Conference on Software Engineering Companion. ACM, 102–111.

[9] Shilin He, Qingwei Lin, Jian-Guang Lou, Hongyu Zhang, Michael R. Lyu, and Dongmei Zhang. 2018. Identifying Impactful Service System Problems via Log Analysis. In Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2018). ACM, 60–70. https://doi.org/10.1145/3236024.3236083

[10] Ke Zhang, Jianwu Xu, Martin Renqiang Min, Guofei Jiang, Konstantinos Pelechrinis, and Hui Zhang. 2016. Automated IT system failure prediction: A deep learning approach.. In BigData. 1291–1300.

[11] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 1285–1298.

[12] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. 2017. Long shortterm memory based operation log anomaly detection. In Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on. IEEE, 236–242.