GLIMPSE - Journal of Computer Science • Vol. 2(2), JULY-DECEMBER 2023, pp.19-24

MALICIOUS URL DETECTION

[1] Rekha Baghel, [2]Tarushi Khanna, [3] Saatvik Rawat, [4]Surbhi Saxena, [5]Mohit Kumar Singh

[1] Assistant Professor, Ajay Kumar Garg Engineering College, Ghaziabad, UP, India

[2,3,4,5] B.Tech 4th year student (CS), Ajay Kumar Garg Engineering College, Ghaziabad, UP, India

[1]BaghelRekha@akgec.ac.in, [2] tarushikhushi@gmail.com, [3]saatvik1911056@akgec.ac.in,

[4]surbhi110601@gmail.com , [5]mohit1911048@akgec.ac.in

Abstract **-The prevalence and danger of network information insecurity are on the rise, and hackers are taking advantage of human vulnerabilities to attack end-to-end technology using techniques such as social engineering, phishing, and pharming. One of the key steps in these attacks is the use of malicious Uniform Resource Locators (URLs) to deceive users. As a result, there has been an increased interest in the detection of malicious URLs using machine learning and deep learning techniques. This paper proposes a method for detecting malicious URLs based on their behaviours and attributes using machine learning algorithms and big data technology. The proposed system consists of a new set of URL features and behaviours, a machine learning algorithm, and big data technology, all aimed at improving the ability to detect malicious URLs based on abnormal behaviours. The experimental results indicate that the proposed URL attributes and behaviour can significantly enhance the ability to detect malicious URLs. Therefore, this proposed system can be considered an optimized and user-friendly solution for detecting malicious URLs.**

Keywords: **URL, malicious URL detection, feature extraction, feature selection, and machine learning.**

I. INTRODUCTION

As society becomes increasingly reliant on online services, the prevalence of online fraud and malicious websites also grows. Many users are unaware of these threats and may assume any website is legitimate, making the prevention of such attacks complex. URLs are particularly vulnerable to malicious attacks, as they are the first and most cost-effective way to access information about websites. Therefore, it is essential to determine whether a URL is malicious or benign.

To identify malicious URLs, various methods are employed. Blacklisting and Heuristic approaches are commonly used, but they are limited in their ability to evolve with the constantly changing threat landscape. Malicious URL detection applications combine static information such as lexical features of the URL string with host information and HTML or JavaScript content to identify malicious URLs [1]. The primary goal of malicious URL detection is to identify and prevent URLs that contain malicious software, phishing attempts, or other harmful content. These URLs can pose significant risks to user safety and privacy, and they can be delivered through various means, such as email, social media, or web pages. A Uniform Resource Locator (URL) is a unique address on the internet that directs visitors to a website and helps them identify and understand its content.

II. PROPOSED METHOD

To refine the method using the available datasets, a model is created that provides a detailed description of the datasets required for training. The foundation of this model is the datasets themselves, as it requires sufficient and accurate data for both malicious and benign URLs. The dataset comprises a list of URLs that have been classified as either malicious or benign. Each URL is characterized by a set of parameters such as the number of dots in the URL, the URL's distance, and token-based diagrams like "google.com."[2] The model is trained using a binary classification technique, also known as the binary regression technique. This approach offers several advantages, such as achieving maximum learning accuracy compared to other machine learning algorithms and requiring less time to learn phishing URLs.

III. ABOUT THE URL

The Uniform Resource Locator, commonly known as URL, is a web address that points to a specific resource on the internet [3]. It provides a way to locate a website and identify what it contains. A URL consists of two parts: the protocol identifier and the resource name. The protocol identifier specifies the method used to retrieve the resource, such as HTTP, FTP, or News. The resource name is the complete address of the resource, which varies depending on the protocol used. For instance, in HTTP, the resource name may contain parameters. The Host parameter denotes the IP address or domain name of the resource, while the File parameter indicates the location of the resource on the machine or host[4]. The Port parameter, which is optional, identifies the port number associated with the resource. Finally, the Query parameter specifies the values and parameters related to the query.

Malicious URL

In addition to blacklisting, another method used to identify malicious URLs is whitelisting. Whitelisting involves creating a database of known benign URLs and allowing access to only those URLs. This technique can be effective in protecting against known threats, but it may restrict

access to legitimate websites that have not been added to the whitelist.

Behavioral analysis is another method used to identify malicious URLs [6]. This technique involves analyzing the behavior of a website to determine whether it is engaging in malicious activities, such as phishing or drive-by downloads [7]. Behavioral analysis can be effective in detecting new threats, but it may also generate false positives if legitimate websites exhibit similar behavior.

Machine learning is another approach to identifying malicious URLs [9]. Machine learning algorithms can be trained to recognize patterns in URLs that are indicative of malicious intent. This approach can be effective in detecting new threats and reducing false positives, but it requires a large amount of data and may be vulnerable to attacks that attempt to evade detection [10]. Ultimately, a combination of these techniques may be used to effectively identify and protect against malicious URLs [11]. It is important for users to remain vigilant and cautious when clicking on links, especially from unfamiliar sources, and to use antivirus software and other security measures to protect against malware and other threats [12].

IV. APPROACH

The blacklisting method entails developing a database containing known malicious URLs, which is then usedto screen incoming URLs. If a URL matches with the blacklisted entries, it is flagged as harmful, and a warning is issued; otherwise, it is presumed safe [12]. Nonetheless, this technique is ineffective in detecting new threats since new malicious URLs emerge every day, which makes it challenging to maintain an exhaustive list of all potential harmful URLs [13]. Blacklisting is an efficient and rapid method that has a low false positive rate, but it has a high false negative rate, meaning that it fails to detect newly generated URLs[14]. An example of a blacklisted URL is Google's Secure Browsing tool, which is used for search engine optimization.

The heuristic-based technique is an advancement of the blacklisting method, which aims to develop a database or a "blacklist of features"[14].This variation of blacklisting involves identifying, extracting, and storing malicious features instead of entire malicious URLs, thereby enabling the detection of threats in more recent URLs. However, it is worth noting that this approach can generate a high number of false positives, leading to inaccurate outcomes. Heuristic-based approaches typically use machine learning techniques to identify features for classification, making it advantageous to use a heuristic-based approach.

V. TRAINING AND TESTING

To implement and validate the malicious URL detection code, certain steps must be taken. Firstly, choose a deployment platform such as a cloud-based service or on-premises server, and ensure that the necessary dependencies and libraries, such as Pandas, Scikit-learn, web frameworks, and APIs, are in place. Next, create a malicious URL detection script that takes an input URL and predicts whether it is malicious or benign, incorporating a pre-trained machine learning model and vectorizer. Define the appropriate validation and error-handling logic to enable URL processing.

It is important to create a set of test cases to cover a variety of scenarios, including benign URLs, known threats, and URLs with unusual or unexpected patterns [15]. The use of automated testing tools like Pytest and Selenium can streamline the testing process. Once testing is complete and any issues have been identified and resolved, the code can be deployed to the production environment to ensure that it performs as expected [16]. Any issues or bugs that arise during testing should also be addressed.

VI. IMPLEMENTATION

The classification model was trained using a dataset comprising of roughly 400,000 URLs obtained from various sources such as Openphish and Alexa whitelists. To ensure the dataset represents both malicious and benign URLs, an 80-20 split between the two was established. In order to train an effective machine learning model, it is crucial to use only the essential features, as using too many features may lead to the model learning from noise and irrelevant patterns [17]. The process of selecting important parameters in the data is called Feature Selection, which can be done by either including important features or excluding irrelevant ones[18]. For this problem, selecting and extracting useful features that describe the URL adequately is the first step. Lexical features, which refer to the textual properties of the URL, were used instead of host and rank features due to their speed, low data storage requirements, and ease of extraction.

Pre-processing techniques such as one hot encoding and bag of words (BoW) were employed in the proposed work. One hot encoding is used to convert categorical data into integer data and to provide more precise predictions than individual labels[19]. BoW is a method oftransforming text into numerical values by counting the occurrences of words and disregarding grammatical details and word order. The CountVectorizer tool was used to tokenize text and create a dictionary of known words, which is also used in machine learning models. TF and TF-IDF were also used to provide insights into the less relevant words in the document. Normalization is employed as the denominator term of the formula, as the length of corpus documents varies[20]. Advanced lexical features were also used to differentiate between malicious and benign URLs by identifying various obfuscation tactics such as the host replacing an IP address or using misspelt URLs. Data

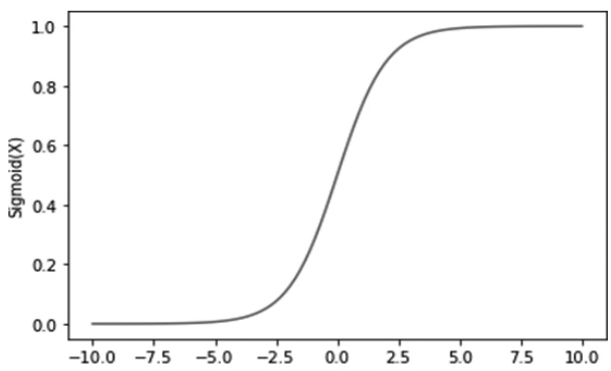
preprocessing is an important step in creating an effective machine learning model. To determine the significance of a word, a dictionary of distinct words needs to be created and its term frequency (TF) needs to be calculated. Commonly encountered words have a higher TF, while rare words have a lower TF. However, the TF does not reflect the importance of words as some words like 'of' and 'and' may be present frequently but are not significant. The weight of each word in the dictionary is determined by its frequency within the corpus.

One limitation of TF-IDF is its failure to accurately capture the semantics of certain words such as "funny" and "humorous," which are synonyms. Moreover, when the vocabulary is large, computing TF-IDF can be computationally expensive.

The Naive-Bayesian classifier algorithm utilizes Bayes' theorem d Assuming that each feature operates independently, the Naive-Bayesian classifier algorithm computes the probability of a given text belonging to a specific outcome class "c" based on the instance "x" that requires classification. The algorithm predicts the tag of the text by selecting the class with the highest probability. This technique is widely employed in Natural Language Processing (NLP).

Random Forest is a supervised machine learning algorithm extensively utilized for solving classification and regression problems. It comprises numerous decision trees, where the internal nodes signify the features of the dataset, branches represent the decision rules, and the leaf nodes symbolize the outcome. Each tree comprises two nodes, namely the Decision Node and the Leaf Node. The Decision Node aids in making decisions and has multiple branches, while the Leaf Node represents the output and does not contain any branches. The decision or test is based on the features of the data. CART is utilized to develop a random forest, which is a classification and regression tree algorithm. The algorithm poses a question and divides the tree into various subtrees based on the answer obtained.

$$TF(w,d) = \frac{occurences\ of\ w\ in\ document\ d}{total\ number\ of\ words\ in\ document\ d}$$



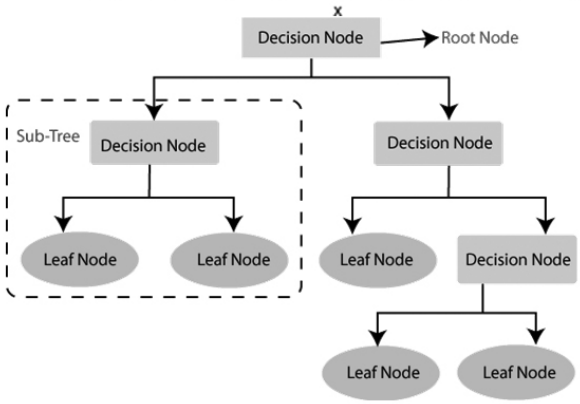


Figure 1(a): Sigmoid function Fig 1(b): Decision Tree

The random forest is a type of classifier that utilizes ensemble learning, which combines several classifiers to improve model performance and tackle complex problems. It comprises a group of decision trees that operate on diverse subsets of the dataset, and the average of these trees boosts the accuracy of predictions. Instead of depending on a single decision tree, the random forest gathers the predictions from each tree and uses the majority vote to determine the final output.

While building the decision tree, pick some random data points from the training set. Then, we'll build the decision trees based on those data points. Finally, we'll find out the predictions for each decision tree and assign them to the categories that get the most votes. We'll repeat the steps 1 and 2.

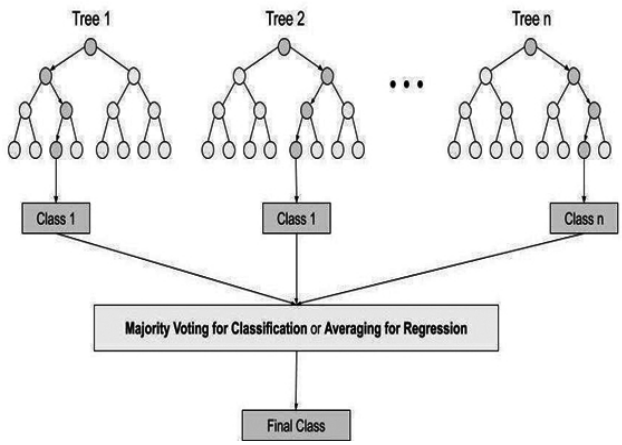


Fig 2: Random Forest

To evaluate model performance, the score() and confusion-matrix techniques were utilized. Scoring is a common method used by Scikit-learn models and estimators to assess accuracy. It requires the input values of the testing sample (X_test) and the expected output values (Y_test) to calculate an accuracy score.

The confusion-matrix is a four-sectioned matrix used to describe the performance of a classification model on a test dataset. These sections are True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

The Classification Report is a metric that evaluates a machine learning model's classification performance, displaying accuracy, the weighted harmonic average of precision and recall, and support. The F1 score measures the model's performance, with a value of 1.0 indicating optimal performance. Accuracy is determined by the ratio of true and false positives of the predicted class, and support is the number of class instances found in the dataset. This report provides an overview of the performance evaluation process, but does not distinguish between models.

This paper describes the two-stage process of detecting malicious URLs using machine learning. The training phase involves the collection of malicious URLs and proper labeling, while the detection phase extracts attributes from each input URL to distinguish between clean and malicious URLs. The dataset is divided into two subsets: the training data used to train machine learning algorithms, and the testing data used to assess model performance. This paper provides a detailed

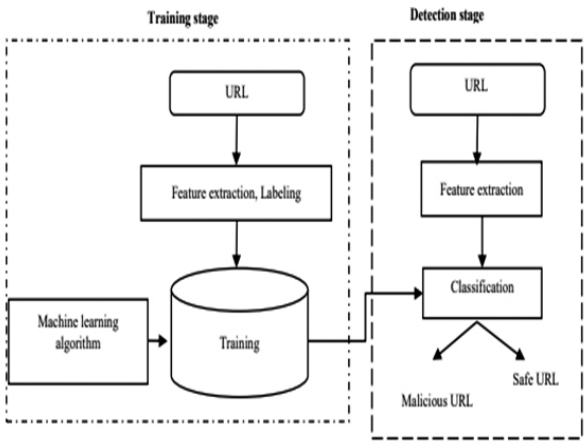


Figure 3(a): Proposed Method

analysis of these attributes. Generating new features from existing ones is an important step in data preparation for statistical analysis. Since machine learning algorithms require numerical inputs, it is necessary to encode URL strings into a numerical vector. By analyzing multiple URLs, it was possible to extract lexical features that can be used to differentiate between good and bad URLs. Tokenization was employed in this example to accomplish this task. The next step involves building a machine learning model (ML model) that represents the output of the training process. This involves training a ML algorithm to predict labels based on the extracted features, fine-tuning the model based on business requirements, and evaluating the model on the holdout data.

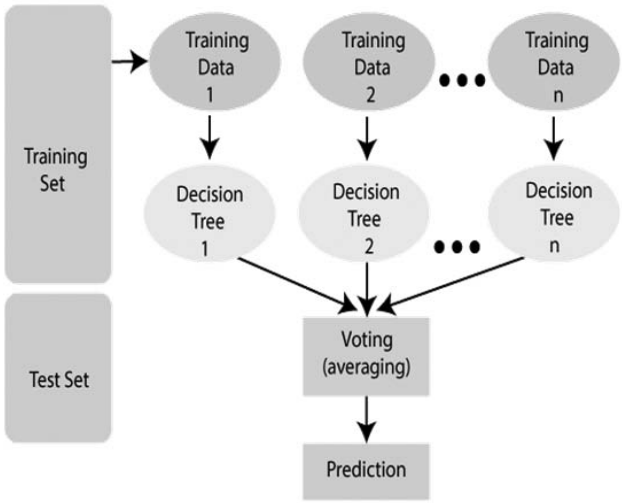


Figure 3 (b): Proposed Method

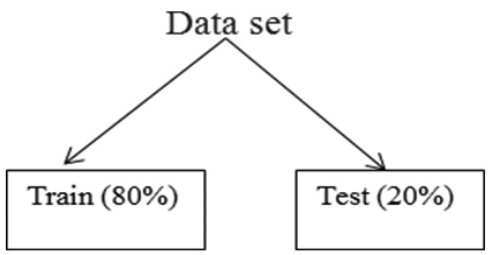


Figure 4: Train and Test Data split

VII. CONCLUSION AND FUTURE ENHANCEMENT

The main objective of this project was to detect malicious URLs using information extracted from the URL string without downloading the page content. Lexical features were extracted using a custom-built function called "Tokenizer", and three machine learning models were evaluated: MultinomialNB, Logistic Regression, and Random Forest. The results of these models were compared using Count-Vectorizedand TF-IdFvectorized data, with Random Forest being the most effective. In the next phase, Fuzzy String matching was used to detect URLs that tried to deceive users by redirecting them to unknown paths.

The application of machine learning techniques to detect malicious URLs is a rapidly expanding area of research, and future work could focus on developing more accurate models through exploring various ML algorithms and feature selection techniques. Real-time detection is also an important consideration, as malicious URLs are constantly evolving, and researchers can explore ways to construct models that can analyze URLs in real-time along with contextual information such as website content and network traffic.

It is important to address ethical considerations when using machine learning for malicious URL detection, including the potential for bias or privacy violations. Adversarial attacks are also a concern, and researchers should explore methods to enhance the resilience of ML models to such attacks.

REFERENCES

[1] Justin Ma, Saul L. K., Savage S., &Voelker G. M.(2011). Learning to detect malicious urls. ACM Transactions on Intelligent Systems and Technology, 3(2), 1–24.

[2] Verma R. & Das A. (2017). Whats in a URL: Fast feature extraction and malicious URL detection. In 3rd International Workshop on Security and Privacy Analytics, pp. 55–63.

[3] Patil D. R. &Patil J. B. (2016). Malicious web pages detection using static analysis of URLs. International Journal of Information Security and Cybercrime, 5(2), 57–70.

[4] Zuhair, H., Selamat, A., &Salleh, M. (2015). Selection of robust feature subsets for phish webpage prediction using maximum relevance and minimum redundancy criterion. Journal of Theoretical and Applied Information Technology, 81(2), 188–205.

[5] HajianNezhad J, Vafaei Jahan M, Tayarani-N M, &Sadrnezhad Z. (2017). Analyzing new features of infected web content in detection of malicious web pages. The ISC International Journal of Information Security, 9(2), 63–83.

[6] Mark Dredze, Koby Crammer, & Fernando Pereira. (2008). Confidence-weighted linear classification. In 25th International Conference on Machine Learning (ICML), pp. 264–271.

[7] Hsu C. W. & Lin C. J. (2002). A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks, 13(2), 415–425. Doi: 10.1109/72.991427.

[8] Crammer K., Dredze M., &Kulesza A. (2009). Multiclass confidence weighted algorithms. In Conference on Empirical Methods in Natural Language Processing, pp. 496–504.

[9] D. Sahoo, C. Liu, S.C.H. Hoi, "Malicious URL Detection using Machine Learning: A Survey". CoRR, abs/1701.07179, 2017.

[10] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: a literature survey," IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp.2091–2121, 2013.

[11] M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive by download attacks and malicious javascript code," in Proceedings of the 19th international conference on World wide web. ACM, 2010, pp. 281–290.

[12] R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," ACM Computing Surveys (CSUR), vol. 48, no. 3, p. 37, 2015.

[13] Internet Security Threat Report (ISTR) 2019–Symantec. https://www.symantec.com/content/dam/symantec/docs/reports/istr-24- 2019-en.pdf [Last accessed 10/2019].

[14] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang, "An empirical analysis of phishing blacklists," in Proceedings of Sixth Conference on Email and Anti-Spam (CEAS), 2009.

[15] C. Seifert, I. Welch, and P. Komisarczuk, "Identification of malicious web pages with static heuristics," in Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian. IEEE, 2008, pp. 91–96.

[16] S. Sinha, M. Bailey, and F. Jahanian, "Shades of grey: On the effectiveness of reputation-based "blacklists"," in Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on. IEEE, 2008, pp. 57–64.

[17] B. Eshete, A. Villafiorita, and K. eldemariam, "Binspect: Holistic analysis and detection of malicious web pages," in Security and Privacy in Communication Networks. Springer, 2013, pp. 149–166.

[18] S. Purkait, "Phishing counter measures and their effectiveness– literature review," Information Management & Computer Security, vol. 20, no. 5, pp. 382–420, 2012.

[19] Y. Tao, "Suspicious url and device detection by log mining," Ph.D. dissertation, Applied Sciences: School of Computing Science, 2014.

[20] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, "Detection of malicious web pages using system calls sequences," in Availability, Reliability, and Security in Information Systems. Springer, 2014, pp. 226–238.

[21] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: an application of large-scale online learning," in Proceedings of the 26th Annual International Conference on Machine Learning. ACM, 2009, pp. 681–688

[22] Leo Breiman.: Random Forests. Machine Learning 45 (1), pp. 5- 32, (2001).

[23] Thomas G. Dietterich. Ensemble Methods in Machine Learning.International Workshop on Multiple Classifier systems, pp 1-15,Cagliari, Italy, 2000 [16]Developer Information. https://www.phishtank.com/developer_info.php.

[24] URLhaus Database Dump. https://urlhaus.abuse.ch/downloads/csv/.

[25] URLhaus Database Dump. https://urlhaus.abuse.ch/downloads/csv/.

[26] Dataset URL. http://downloads.majestic.com/majestic_million.csv.

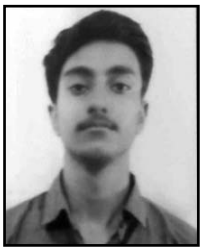
[27] Malicious_n_Non-MaliciousURL. https://www.kaggle.com/antonyj453/ url dataset#data.csv.

ABOUT THE AUTHOR

Ms. Rekha Baghel got her B.E degree from IET Agra in 2008.She has done her M. Tech from NIT Jalandhar, India. She is pursuing PhD from IGDTUW, New Delhi. She has around 11 years of experience. She is an Assistant Professor in Department of Computer Science & Engineering at AKGEC. She is doing research in Machine Learning with an objective of providing various problem solution to society. Her areas of Interest include Machine Learning, Deep learning, Swarm Intelligence.

Mr. Mohit Kumar Singh was born and brought up in Badaun .He is currently pursuing a Bachelor's degree in Engineering (B.Tech) in computer science and engineering from Ajay Kumar Garg Engineering College. He completed his schooling from Blooming Dale School Budaun, Uttar Pradesh. He loves working with new people and He is empathetic and well coordinated, this makes

him a team player. He has always been interested in computers and UI/UX . He has completed an internship in the same domain and from an alumni startup Ambulon, Gzb. Currently He is working as a trainee at Farmtaste and building his side hustle in forms of drop shipping and other creative business.



Mr. Saatvik Rawat was born and brought up in Muzaffarnagar. He is currently pursuing a Bachelor's degree in Engineering (B.Tech) in computer science and engineering from Ajay Kumar Garg Engineering College. He has completed his schooling from S.D. Public School, Muzaffarnagar. He has a strong knowledge of programming languages and software development. He is an enthusiastic App developer using the Flutter framework. He has strong knowledge of Data Structures and algorithms.

Ms. Surbhi Saxena was born and brought up in Ghaziabad. She is currently pursuing a Bachelor's degree in Engineering (B.Tech) in computer science and engineering from Ajay Kumar Garg Engineering College. She has completed my schooling from Vivekanand School, Delhi. She believes that her attention to detail and willingness to learn new skills complement each other exceptionally well. She has keen interest in machine learning and has completed an internship in the same domain and from DRDO, Delhi. Currently She is working as a cloud intern at PWC.

Ms. Tarushi Khanna was born and brought up in Ghaziabad. She is currently pursuing a Bachelor's degree in Engineering (B.Tech) in computer science and engineering from Ajay Kumar Garg Engineering College. She has completed my schooling from St. Francis School, Indirapuram. She has strong programming skills and a keen interest in machine learning. She has completed two internships in ML and her most recent one is from DRDO, Delhi.