

Detecting Phishing Attacks using Machine Learning

A project report submitted

to

MANIPAL ACADEMY OF HIGHER EDUCATION

For Partial Fulfillment of the Requirement for the

Award of the Degree

of

Bachelor of Technology

in

Information Technology

by

Ganesh S Nayak

Reg. No. 200911008

Under the guidance of

Dr. Balachandra

Professor

Department of I & CT

Manipal Institute of Technology

Manipal, India



MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent unit of MAHE, Manipal)

May 2024

I dedicate this thesis to express profound appreciation to my cherished friends and family, whose boundless love and support have illuminated my path. To my beloved parents, whose enduring dedication and sacrifices have been the bedrock of my academic journey, I owe immeasurable gratitude.

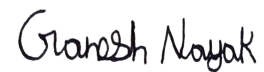
Furthermore, I extend sincere thanks to Dr. Balachandra, esteemed Professor at the Department of I&CT, whose unwavering guidance and encouragement have been pivotal in sculpting the trajectory of my research. His guidance has not only enhanced my academic journey but also instilled confidence in my capabilities. I am profoundly grateful to him for the invaluable knowledge he has selflessly imparted during this endeavor.

DECLARATION

I hereby declare that this project work entitled **Detecting Phishing Attacks using Machine Learning** is original and has been carried out by me in the Department of Information and Communication Technology of Manipal Institute of Technology, Manipal, under the guidance of **Dr. Balachandra, Professor**, Department of Information and Communication Technology, M. I. T., Manipal. No part of this work has been submitted for the award of a degree or diploma either to this University or to any other Universities.

Place: Manipal

Date :10-05-24



Ganesh S Nayak



MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

CERTIFICATE

This is to certify that this project entitled **Detecting Phishing Attacks using Machine Learning** is a bonafide project work done by **Mr. Ganesh S Nayak (Reg.No.:200911008)** at Manipal Institute of Technology, Manipal, independently under my guidance and supervision for the award of the Degree of Bachelor of Technology in Information Technology.

Dr.Balachandra

Professor

Department of I & CT

Manipal Institute of Technology

Manipal, India

Dr.Smitha N Pai

Professor & Head

Department of I & CT

Manipal Institute of Technology

Manipal, India

ACKNOWLEDGEMENTS

I extend my profound gratitude to all those whose invaluable contributions have been instrumental in the realization of my final semester project. At the forefront, I extend my sincerest gratitude to my internal mentor, Dr. Balachandra, for his indispensable guidance, mentorship, and steadfast support throughout this endeavor. His profound expertise and unwavering commitment have significantly enriched the depth and quality of my research. Furthermore, I extend my sincere thanks to Ms. Smitha N Pai, the Head of the I&CT Department, for affording us the opportunity to conduct my project within the department and for facilitating the research process seamlessly. Additionally, I acknowledge Dr. Anil Rana, the Director of MIT, Manipal, for fostering an environment conducive to scholarly inquiry and for furnishing me with the necessary resources. Lastly, I am deeply indebted to my cherished friends and family members for their enduring encouragement, assistance, and camaraderie, all of which have infused this academic pursuit with enduring value. Without the collective support and guidance of these individuals, the successful completion of this project would have remained beyond reach.

ABSTRACT

Phishing attacks pose a significant threat in today's interconnected digital landscape, targeting individuals and organizations worldwide. These deceptive tactics involve fraudulently obtaining sensitive information through disguised websites, leading to financial losses and breaches of privacy. Effective phishing detection is crucial for mitigating risks and protecting against malicious cyber activities. This project leverages machine learning techniques to detect phishing websites and enhance cybersecurity defenses.

The project adopts a comprehensive approach to phishing detection, leveraging a dataset comprising 111 features extracted from the latest phishing websites. The primary objective is to discern patterns between phishing and legitimate websites, facilitating the development of robust detection models. A novel feature importance framework streamlines the feature selection process and improves detection efficiency. Various machine learning classifiers, including Feedforward, DNN, Wide and Deep, and TabNet architectures, are utilized to evaluate the effectiveness of different models in detecting phishing websites.

Through rigorous experimentation and grid search optimization of hyperparameters, the project identifies the optimal model configuration for robust phishing detection. Notably, the Feedforward model achieves a remarkable accuracy rate of 94% using only 14 key features, underscoring the capabilities of machine learning in identifying phishing attempts. Additionally, the project introduces an innovative evaluation metric termed the 'anti-phishing score,' providing a comprehensive assessment of model effectiveness. Overall, the project's findings contribute to enhancing cybersecurity defenses against phishing attacks, providing valuable insights for practitioners and researchers.

[Computing methodologies]: Machine learning - Machine learning approaches - Neural networks.

[Security and privacy]: Intrusion/anomaly detection and malware mitigation - Social engineering attacks - Phishing.

Contents

| | |
|--|-----------|
| Acknowledgements | iv |
| Abstract | v |
| List of Tables | ix |
| List of Figures | xi |
| Abbreviations | xi |
| Notations | xiii |
| 1 Introduction | 1 |
| 1.1 Area of Work | 1 |
| 1.2 Motivation | 1 |
| 1.3 Objective | 2 |
| 1.4 Importance of the End Result | 3 |
| 2 Literature Review | 4 |
| 2.1 Literature Review | 4 |
| 3 Materials and Proposed Methods | 10 |
| 3.1 Datasets | 10 |
| 3.2 Tools | 11 |
| 3.2.1 Python | 11 |
| 3.2.2 Google Colab | 11 |

| | | |
|----------|---|-----------|
| 3.2.3 | TensorFlow and Keras | 11 |
| 3.2.4 | Scikit-learn (sklearn) | 12 |
| 3.2.5 | Proposed DL Methods | 12 |
| 3.2.5.1 | The Feedforward Neural Network (FNN) Model | 13 |
| 3.2.5.2 | The Deep Neural Network (DNN) Model | 15 |
| 3.2.5.3 | The Wide and Deep Model (WDM) | 16 |
| 3.2.5.4 | The TabNet Model | 18 |
| 4 | Methodology | 20 |
| 4.0.1 | Data Collection | 20 |
| 4.0.2 | Preprocessing | 22 |
| 4.0.3 | Feature Selection | 22 |
| 4.0.4 | Feature Importance | 22 |
| 4.0.5 | Train and Test Model | 23 |
| 4.0.6 | Evaluation | 24 |
| 5 | Results | 25 |
| 5.1 | Data Cleaning and Data Transformation | 25 |
| 5.1.1 | Data Cleaning and Consistency | 25 |
| 5.1.2 | Data Transformation: Min-Max Normalization | 26 |
| 5.2 | Feature Selection | 27 |
| 5.2.1 | Correlation-based Feature Selection | 27 |
| 5.2.2 | Feature Importance | 27 |
| 5.3 | Model Implementation | 32 |
| 5.4 | Model Training and Evaluation | 33 |
| 5.4.1 | Model Training | 33 |
| 5.4.2 | Model Evaluation | 36 |
| 5.5 | Optimizing Model Performance: Grid Search Hyperparameter Tuning . | 38 |
| 5.6 | Automated Script | 40 |

| | |
|--|-----------|
| 6 Conclusion and Future Scope | 41 |
| 6.1 Work Summary | 41 |
| 6.2 Conclusions | 41 |
| 6.3 Future Scope of Work | 42 |
| Appendices | 44 |
| A Code | 45 |
| A.1 Architecture of Feedforward Neural Network (FNN) Model | 45 |
| A.2 Architecture of DNN (Deep Neural Network) Model | 45 |
| A.3 Architecture of Wide & Deep Model | 46 |
| A.4 Architecture of TabNet Classifier Model | 46 |
| References | 48 |
| Project Detail | 51 |

List of Tables

| | | |
|-----|---|----|
| 3.1 | DL Hyperparameters for Training the Datasets. | 13 |
| 5.1 | Top Features Selected through Permutation Importance Analysis | 31 |
| 5.2 | Top Features Selected through Permutation Importance Analysis | 32 |
| 5.3 | Model Performance Metrics | 36 |
| 5.4 | Hyperparameter Grid Search Results | 39 |
| A.1 | Project Detail | 52 |

List of Figures

| | | |
|-----|--|----|
| 3.1 | Python logo | 11 |
| 3.2 | Google Colab logo | 11 |
| 3.3 | TensorFlow and Keras logos | 12 |
| 3.4 | Scikit-learn logo | 12 |
| 3.5 | Architecture of the Feedforward Neural Network (FNN) model. | 15 |
| 3.6 | Architecture of the Deep Neural Network (DNN) model. | 17 |
| 3.7 | Architecture of the Wide and Deep model. | 18 |
| 3.8 | Architecture of the TabNet model. | 19 |
| 4.1 | Flow diagram illustrating the proposed classification system for fall de- tection based on machine learning. | 21 |
| 4.2 | An Example of URL Structure. | 21 |
| 4.3 | Flowchart illustrating the Permutation Importance-based Feature Selec- tion process in the machine learning pipeline. | 23 |
| 4.4 | Evaluation Workflow for the Feature Selection Approach. | 24 |
| 5.1 | Number of Columns Before and After Data Cleaning. | 26 |
| 5.2 | The Permutation Importance Bar chart and Ranking depict the signifi- cance of features in the Feedforward Neural Network Model. | 29 |
| 5.3 | The Permutation Importance Bar chart and Ranking depict the signifi- cance of features in the Deep Neural Network Model. | 29 |
| 5.4 | The Permutation Importance Bar chart and Ranking depict the signifi- cance of features in the Wide and Deep Model. | 30 |

| | | |
|------|---|----|
| 5.5 | The Permutation Importance Bar chart and Ranking depict the significance of features in the TabNet Model. | 30 |
| 5.6 | Feedforward Neural Network Model Evaluation Results Across Ten-Fold Cross Validation. | 34 |
| 5.7 | Deep Neural Network Model Evaluation Results Across Ten-Fold Cross Validation. | 34 |
| 5.8 | Wide & Deep Model Evaluation Results Across Ten-Fold Cross Validation. | 35 |
| 5.9 | TabNet's Evaluation Results Across Ten-Fold Cross Validation. | 35 |
| 5.10 | ROC Curves | 37 |
| 5.11 | Precision-Recall Curves | 38 |

ABBREVIATIONS

| | | |
|------|---|-------------------------------------|
| ML | : | Machine Learning |
| API | : | Application Programming Interface |
| FNN | : | Feedforward Neural Network |
| PCA | : | Principal Component Analysis |
| DNN | : | Deep Neural Network |
| LDA | : | Linear Discriminant Analysis |
| KNN | : | K-Nearest Neighbors |
| KPCA | : | Kernel Principal Component Analysis |
| WDM | : | Wide and Deep Model |
| SVD | : | Singular Value Decomposition |
| RNN | : | Recurrent Neural Network |
| ICA | : | Independent Component Analysis |
| CNN | : | Convolutional Neural Network |
| WDM | : | Wavelength Division Multiplexing |
| TTL | : | Time To Live |
| TPR | : | True Positive Rate |
| TLD | : | Top-Level Domain |
| FPR | : | False Positive Rate |
| TLS | : | Transport Layer Security |
| ROC | : | Receiver Operating Characteristic |

NOTATIONS

| | | |
|-------------------------|---|---|
| $X_{\text{normalized}}$ | : | Normalized value of feature X |
| r | : | Pearson correlation coefficient |
| X | : | Initial value of the feature |
| X_{\min} | : | Smallest feature value within the dataset |
| X_{\max} | : | Largest feature value within the dataset |
| X_i | : | Individual value of the feature X |
| \bar{X} | : | Average value of the feature X |
| Y_i | : | Specific value of the feature Y |
| \bar{Y} | : | Average value of the feature Y |

Chapter 1

Introduction

1.1 Area of Work

In the interconnected digital landscape of the 21st century, cybersecurity has emerged as a critical concern, with phishing attacks posing significant threats to individuals, businesses, and organizations worldwide. Phishing, a deceptive practice involving the fraudulent acquisition of sensitive information, jeopardizes cybersecurity, privacy, and financial stability. With the proliferation of cybercrime, the importance of effective phishing detection has become paramount in safeguarding against malicious cyber activities. As technology continues to advance and cyber threats become increasingly sophisticated, the need for robust cybersecurity measures to combat phishing attacks becomes more urgent. Furthermore, the interconnected nature of digital platforms and the widespread adoption of online communication channels have created new avenues for cybercriminals to exploit, emphasizing the critical role of proactive phishing detection in today's cybersecurity landscape.

1.2 Motivation

Despite advancements in cybersecurity measures, phishing attacks continue to evolve in sophistication, exploiting vulnerabilities across various online platforms, including email, social media, and messaging services [1,2]. Previous studies have highlighted the

prevalence of phishing attacks and underscored the need for robust detection mechanisms capable of identifying evolving tactics [3,4]. Recognizing the severe consequences of falling victim to such attacks, including financial losses, identity theft, and reputational damage [1,2], it becomes evident that enhancing phishing detection methodologies is crucial. Moreover, phishing attacks exacerbate cybersecurity challenges by leading to data breaches and other cybercrimes [1,2]. Therefore, innovation in phishing detection methodologies is imperative to stay ahead of cybercriminals who continually adapt their tactics to exploit emerging vulnerabilities. The research builds upon existing studies to investigate machine learning-based approaches for detecting phishing emails [2–6]. By evaluating multiple machine learning models through rigorous experimentation and analysis, including feed-forward, DNN, Wide and Deep, and TabNet architectures, the aim is to provide valuable insights into their strengths, limitations, and practical implications. This research contributes to ongoing efforts to enhance cybersecurity measures and protect users from the pervasive threat of phishing in the digital age.

1.3 Objective

The primary objective of this research is to:

1. Assess the efficacy of machine learning techniques in identifying phishing emails.
2. Systematically evaluate and compare multiple machine learning models, drawing insights from existing literature and empirical data.
3. Offer insightful analysis into the advantages, constraints, and practical applications of different machine learning models for phishing detection.
4. Conduct rigorous experiments and analysis to enhance cybersecurity defenses and mitigate the pervasive threat of phishing in today’s digital landscape.

5. Investigate novel approaches and techniques to enhance the precision and effectiveness of phishing detection systems.
6. Address the evolving nature of phishing attacks and adapt detection techniques accordingly.
7. Play a role in fortifying cybersecurity frameworks by utilizing research outcomes and knowledge to develop stronger defense mechanisms.

1.4 Importance of the End Result

The significance of the research lies in its potential to enhance cybersecurity measures and protect users from the detrimental impact of phishing attacks. By developing and optimizing machine learning models for phishing detection, the research contributes to ongoing efforts in enhancing cybersecurity resilience and safeguarding sensitive information. The end result of this research is expected to provide valuable insights and practical solutions for combating phishing threats, thereby bolstering cybersecurity defenses and fostering a safer digital environment for individuals and organizations alike. Furthermore, the findings of this research have implications beyond cybersecurity, as successful phishing detection can help preserve trust and confidence in digital platforms, thereby facilitating secure online interactions and transactions. Ultimately, the research outcomes aim to empower stakeholders across various sectors to better defend against phishing attacks and mitigate their potential consequences.

Chapter 2

Literature Review

2.1 Literature Review

In [1], Salahdine, El Mrabet, and Kaabouch investigate effective phishing attack detection, focusing on the threat of social engineering attacks on users' email accounts. They employ a machine learning-based approach, analyzing over 4000 phishing emails to showcase the effectiveness of an artificial neural network. While achieving robust detection metrics, a notable limitation arises from the exclusive reliance on a single algorithm, underscoring the need for future research to explore alternative models.

In [2], Baykara and Gürel make a notable contribution to cybercrime prevention by introducing the "Anti Phishing Simulator." The unveiling took place during a symposium focused on digital forensic and security, held in 2018. Leveraging a Bayesian algorithm, the tool detects phishing and spam emails by analyzing content, highlighting the severity of phishing attacks and underscoring the significance of devising effective countermeasures.

In [3], Chiew, Yong, and Tan conduct a thorough examination, published in "Expert Systems with Applications," concerning the evolving landscape of phishing attacks. They systematically explore past and current phishing methodologies, discussing asso-

ciated mediums and vectors. The survey enhances understanding of existing phishing techniques, laying the groundwork for devising a comprehensive solution against phishing and directing future research endeavors in classifying phishing types.

In [4], Yahya et al. confront the surge in cyberattacks, notably phishing, heightened during the global reliance on digital operations amid the COVID-19 pandemic. Their study, presented during the Data Science and Its Applications Conference of 2021, delves into machine learning techniques for detecting phishing websites. Utilizing three supervised learning models—Random Forest, Decision Tree, and K-Nearest Neighbour (KNN)—the research underscores the significance of carefully selecting models and encourages further exploration with different datasets for comparative analysis.

In [5], Alswailem, Alabdullah, Alrumayh, and Alsedrani address the critical concern of phishing websites, emphasizing human vulnerabilities. Their work introduces an intelligent browser extension for detecting phishing websites, leveraging machine learning, specifically the Random Forest technique. The research achieves an impressive 98.8% accuracy rate through 26 carefully chosen features, contributing significantly to robust machine learning-based solutions for identifying and mitigating phishing threats.

In [6], Zuhair, Selamat, and Salleh perform a literature review focusing on challenges associated with deceptive websites and advocating for improved phishing detection. The study emphasizes the integration of feature selection methods, machine learning classifiers, and hybrid features. Persistent limitations in classification performance across the web are noted due to challenges in selecting optimal features. The review critically assesses research efforts, identifies limitations, and proposes avenues for improvement, guiding future developments in phishing detection for researchers.

In [7], Chinnasamy et al. tackle the pervasive threat of phishing attacks within the context of a conference focused on progress in intelligent, secure, and smart computing

conducted in 2022. They introduce an effective methodology for identifying phishing attacks using machine learning algorithms, presenting a heuristic approach for classification based on features such as web traffic and URL. Recognizing the limitations of existing anti-phishing approaches, their research aims to overcome zero-day phishing challenges, offering a comprehensive solution to detect and mitigate phishing attacks.

In [8], T. R. N and R. Gupta delve into the pivotal significance of feature selection in artificial intelligence and machine learning. Presented during an international conference hosted by the IEEE in 2020, their study emphasizes the need to eliminate redundant features for reduced computational expenses and improved classifier accuracy. This investigation offers valuable perspectives on various feature selection methodologies, influencing the evolving domain of machine learning. It highlights the critical role of thoughtful feature selection in enhancing model predictability, offering essential insights for both researchers and practitioners.

In [9], Ramachandran, Ravichandran, and Raveendran discuss challenges related to high dimensionality in big data. They present their research on dimensionality reduction techniques during the conference held in 2020. The research evaluates methods like PCA, LDA, KPCA, SVD, and ICA, providing a comparative analysis of their effectiveness in reducing redundancy and simplifying attribute sets in large datasets. This work is a valuable reference for understanding dimensionality reduction complexities in handling extensive datasets.

In [10], Dangwal and Moldovan enhance machine learning-driven phishing detection by consolidating datasets and applying feature selection to identify 13 optimal features. Comparative evaluation reveals that models utilizing all features exhibited decreased performance when tested with the dataset containing 30 features but showcased superior results on the 48-feature dataset. The leading model, leveraging the selected 13 features, attained an accuracy of 0.937. This study underscores the significance of

feature selection in enhancing machine learning-driven phishing detection endeavors.

In [11], Chinguwo and Dhanalakshmi explore phishing detection in cloud environments using a stacking ensemble machine learning technique. Achieving 98.8% accuracy, they analyze features like address bar and domain characteristics. This study underscores evolving phishing detection through machine learning. Evaluation against prior studies highlights efficacy in addressing cloud security challenges. The study contributes to the growing body of literature on combating cyber threats in cloud computing.

In [12], Rajeswary and Thirumaran introduce an LSTM-driven automated model designed to detect various attacks on Tor hidden services. Their approach integrates random forest, CNN, and LSTM models to achieve high accuracy in spotting phishing attempts within the Tor network. The authors propose a method specifically targeting vulnerable attacks in phishing-based Tor hidden services, employing LSTM and random forest classifiers. Their model achieves a maximum detection accuracy of 95.60% and 95.77% on the CIRCL and AIL datasets, respectively, enhancing security in online environments.

In [13], Basant Subba presents a novel security framework based on a diverse stacking ensemble for phishing attack detection. This framework integrates multiple classifiers, achieving high accuracies on benchmark datasets and surpassing existing frameworks. Notably, it leverages 24 URL-based and 20 web-page based features, showcasing a comprehensive approach to phishing attack detection. By integrating diverse classifiers within a stacking ensemble framework, this study advances security measures against phishing threats, addressing the evolving landscape of cyber attacks in online environments.

In [14], Peter Švančárek discusses phishing attack detection using machine learning techniques, emphasizing the efficacy of Random Forest in distinguishing phishing web-

sites. The chapter highlights the successful utilization of Random Forest classifiers in identifying phishing websites, particularly in sectors like e-commerce and financial management where phishing poses significant risks. This underscores the growing importance of machine learning techniques, especially Random Forest, in combating the pervasive threat of phishing attacks.

In [15], Diana T. Mosa, Mahmoud Y. Shams, Amr A. Abohany, El-Sayed M. Elkenawy, and Mostafa Thabet investigate machine learning methods for identifying phishing URL attacks, achieving accuracies ranging from 90.23% to 95.43%. Their study provides a thorough examination of contemporary machine learning approaches to tackle phishing challenges, demonstrating high accuracy levels. With a dataset comprising 11,000 websites, the research highlights the effectiveness of these methods in countering phishing threats, offering valuable insights into the intersection of cybersecurity and machine learning in addressing online risks.

In [16], Shusheng Yu et al. introduce an approach to phishing detection based on a neural network model with multiple features, merging MLP, CNN, and RNN. Their method achieves high accuracy and recall rates, outperforming other algorithms. By employing methods for extracting multiple features and deep learning techniques, the approach incorporates MLP for self-defined feature extraction, CNN for image feature extraction, and RNN for text feature extraction. The fused feature vectors enable effective detection of phishing attacks.

In [17], Jasmina Novakovic and Suzana Marković propose a method to detect phishing attacks based on URLs, employing neural network models. Their study focuses on utilizing neural networks for binary classification, aiming to achieve high accuracy in distinguishing between legitimate and fraudulent URLs. By evaluating model performance using binary classification accuracy, the research contributes to enhancing phishing detection by applying machine learning techniques to combat fraudulent ac-

tivities in network environments.

In [18], Shakirat Aderonke Salihu et al. propose a heuristic-based method for detecting phishing URLs. Their approach achieves an 85% true positive rate and 95.52% accuracy by utilizing attributes extracted from PhishTank and Alexa data. The authors introduce a novel method involving attribute extraction, filtering, and classification based on their impact on a website. Leveraging heuristic features, the study advances phishing detection techniques, utilizing data from PhishTank and Alexa to implement a Python model for high accuracy in identifying phishing URLs.

In [19], Jibrilla Tanimu and Stavros Shiaeles present a new approach leveraging machine learning for addressing the shortcomings of current phishing detection methods. Their model focuses on image visualization of website code and feature extraction from malicious URLs to enhance efficacy and accuracy. By exploring existing method drawbacks, the authors highlight the need for innovative solutions to combat evolving phishing threats. The proposed model aims to improve detection accuracy and efficacy, advancing cybersecurity.

In [20], Manuel Sánchez-Paniagua et al. present a technique for identifying fraudulent websites utilizing an innovative dataset and web-related features. Their approach employs a LightGBM classifier and 54 web technology features on the PILWD, achieving 97.95% accuracy. The study addresses limitations in current anti-phishing techniques and demonstrates how web technology features enhance detection accuracy. This methodology promises to enhance cybersecurity measures against phishing attacks.

Chapter 3

Materials and Proposed Methods

3.1 Datasets

In the initial phase of the research, the primary focus was on acquiring diverse datasets to support the experimentation. One of the pivotal datasets utilized was sourced from Mendeley 2020, which is accessible at the following link: <https://data.mendeley.com/datasets/72ptz43s9v/1>. This dataset served as a fundamental resource for the investigation. Through meticulous analysis, the structure of URLs was examined, identifying five key components: Directory, Parameters, Domain, File, and the complete URL. The contributors to the Mendeley 2020 dataset meticulously extracted features from URLs, allowing for detailed analysis. These features encompassed various signs across each component, providing valuable insights for the research.

Additionally, another dataset, Mendeley 2021, was incorporated into the experimentation process. This dataset, available at <https://data.mendeley.com/datasets/c2gw7fy2j4/1>, played a crucial role in validating and testing the models during the hyperparameter tuning phase. By integrating these new samples into the analysis, the aim was to enhance the robustness and effectiveness of the research findings.

3.2 Tools

3.2.1 Python

Python was the main programming language utilized in the project due to its versatility, rich libraries, and user-friendly nature, making it well-suited for the development and implementation of both machine learning and deep learning models. Python was leveraged for data preprocessing, model development, experimentation, and result analysis.



Figure 3.1: Python logo

3.2.2 Google Colab

Google Colab provided a powerful cloud-based platform for running Python code. It granted free access to GPU and TPU resources, facilitating the efficient training of sophisticated machine learning models. The collaborative nature of Google Colab enabled seamless collaboration among team members by facilitating real-time sharing and editing of notebooks.



Figure 3.2: Google Colab logo

3.2.3 TensorFlow and Keras

TensorFlow and Keras were essential resources for constructing and training deep learning models. TensorFlow, a machine learning framework created by Google, offered a versatile platform for building intricate neural networks. Meanwhile, Keras,

a user-friendly high-level API built on TensorFlow, streamlined the development and deployment of neural networks. These resources were utilized to deploy a variety of deep learning architectures, including feedforward neural networks (FNNs), deep neural networks (DNNs), TabNet, and Wide and Deep models.



Figure 3.3: TensorFlow and Keras logos

3.2.4 Scikit-learn (sklearn)

The renowned Python machine learning library, Scikit-learn, provided a wide array of functionalities for data preparation, model choice, and assessment. It was utilized for tasks like feature scaling, cross-validation, and performance metric computation. Its intuitive interface and detailed documentation were instrumental in building both traditional machine learning models and deep learning architectures.



Figure 3.4: Scikit-learn logo

These tools collectively enabled the streamlining of the workflow, from data pre-processing to model development and evaluation. Their integration facilitated efficient experimentation and iteration, ultimately contributing to the success of the project.

3.2.5 Proposed DL Methods

In this research endeavor, four distinct deep learning (DL) models were employed for phishing detection tasks. These models encompassed a Feedforward Neural Network

(FNN), a Deep Neural Network (DNN), a Wide and Deep Model, and a TabNet Model.

The FNN and DNN designs included several fully connected layers featuring Rectified Linear Unit (ReLU) activation functions, with a sigmoid activation function at the output layer. Similarly, the Wide and Deep Model amalgamated both wide and deep components using a functional API in TensorFlow Keras. The wide component excelled at memorizing intricate details from data, while the deep component uncovered complex patterns. The WDM leveraged these strengths for robust classification.

Furthermore, the TabNet Model, a novel DL architecture integrating sequential attention mechanisms for tabular data, was employed. Pre-training of the TabNet Model was conducted using a TabNet Pretrainer, followed by fine-tuning for classification tasks.

Hyperparameter tuning was performed for each DL model to optimize their performance. Considered hyperparameters encompassed dropout rates, epochs, batch size, and early stopping criteria. Grid search optimization was utilized to determine the most suitable hyperparameters for every model structure.

Table 3.1: DL Hyperparameters for Training the Datasets.

| Parameter | Value |
|-------------------|---------------------|
| Epochs | 10 |
| Loss Function | Binary Crossentropy |
| Batch Size | 64 |
| Optimizer | Adam |
| Hidden Activation | ReLU |
| Output Activation | Sigmoid |
| Metrics | Accuracy |

3.2.5.1 The Feedforward Neural Network (FNN) Model

In the realm of phishing detection, the Feedforward Neural Network (FNN) is a cornerstone in deep learning methodologies, celebrated for its adaptability and wide-

ranging application across various classification tasks. Unlike Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs), FNNs process input data sequentially, free from cyclic dependencies, making them especially suitable for tasks where temporal order holds lesser significance.

The architecture of the FNN model encompasses three interconnected layers: an input layer, a hidden layer, and an output layer (as depicted in Figure 3.5). The input layer accepts the feature vector representing preprocessed email data, typically containing vital information pertinent to phishing detection. The quantity of neurons within the input layer aligns with the dimensionality of the feature vector, contingent upon the nature and representation of the selected features.

The central computational operations of the FNN model occur in its hidden layer(s) (see Figure 3.5), where it diligently extracts significant features from input data through weighted links and utilizes non-linear activation functions. In the implementation, Rectified Linear Unit (ReLU) activation functions grace the hidden layers, bestowing upon the model the ability to discern intricate data patterns efficiently through introduced non-linearity.

At the model's conclusion lies the output layer, where final predictions or classifications are unveiled (refer to Figure 3.5). Tailored for phishing detection, this layer comprises a solitary neuron, employing a sigmoid activation function to yield a probability score indicative of the likelihood of a sample constituting a phishing attempt. The sigmoid function ensures that resultant probabilities span the interval $[0, 1]$, facilitating intuitive interpretation. Training the FNN model is orchestrated through the Adam optimizer, esteemed for its efficacy and resilience in gradient-based optimization tasks. Binary cross-entropy emerges as the preferred loss function during training, aptly suited for binary classification endeavors like phishing detection, measuring the discordance between predicted probabilities and true class labels, guiding the model towards parameter optimization. Furthermore, performance evaluation of the FNN model ensues through the prism of accuracy, quantifying the proportion of accurately classified samples amidst the entire dataset. By carefully fine-tuning hyperparameters

and refining architectural nuances, the objective is to engender an FNN-based phishing detection system, characterized by robustness and efficacy, adept at delineating between phishing emails and legitimate communications with utmost precision and efficiency.

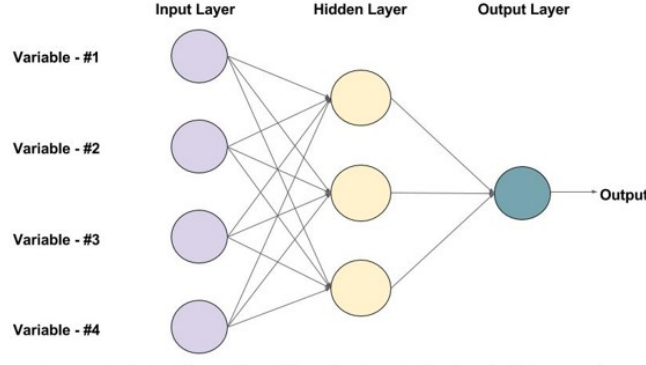


Figure 3.5: Architecture of the Feedforward Neural Network (FNN) model.

3.2.5.2 The Deep Neural Network (DNN) Model

This model reigns supreme as a cornerstone in the realm of deep learning methodologies. Renowned for its versatility and efficacy in tackling diverse classification tasks, DNNs demonstrably excel in phishing detection. As depicted in the accompanying figure (Figure 3.6), DNNs leverage a multilayered architecture to progressively extract intricate patterns and features from input data.

The DNN model is meticulously architected with a series of densely interconnected layers (refer to Figure 3.6), progressively extracting salient information from input data. These hidden layers, equipped with a multitude of interconnected neurons, act as the engine room of the model, meticulously orchestrating complex computations to unearth meaningful patterns within the data. Each hidden layer is endowed with Rectified Linear Unit (ReLU) activation functions, introducing essential non-linearity and enabling the model to efficiently learn intricate data representations. The ReLU activation function, represented mathematically as $f(x) = \max(0, x)$, ensures that only non-negative values are forwarded, thereby boosting the model's ability to capture

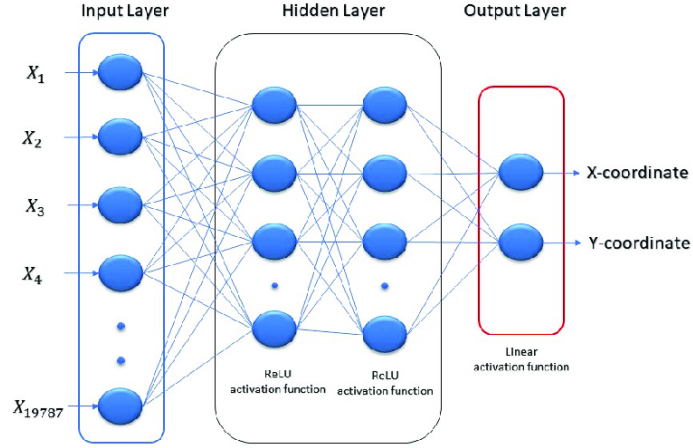
intricate relationships within the dataset.

At the topmost layer of the DNN architecture, as illustrated in Figure 3.6, lies the output layer, responsible for revealing the final predictions or classifications. Concerning phishing detection, this layer includes a solitary neuron utilizing a sigmoid activation function, which generates a probability score ranging from 0 to 1. This score signifies the likelihood of a sample being a phishing attempt. The sigmoid function guarantees that the output probability falls within the specified range, enabling straightforward and understandable interpretation.

Training of the DNN model is orchestrated through the Adam optimizer, a robust and efficient gradient-based optimization algorithm. The choice of optimizer is crucial in guiding the model towards optimal parameter values, thereby enhancing its predictive performance. Binary crossentropy emerges as the preferred loss function during training, well-suited for binary classification tasks like phishing detection. Binary crossentropy measures the disparity between predicted probabilities and true class labels, offering crucial insights to the model during optimization. The effectiveness of the DNN model is primarily assessed through accuracy, a pivotal metric in gauging its performance. Accuracy measures the proportion of correctly classified samples amidst the entire dataset, offering insights into the model's predictive prowess. By meticulously fine-tuning hyperparameters and optimizing architectural nuances, the endeavor is to cultivate a robust and reliable phishing detection system, underpinned by the efficacy and versatility of the DNN model.

3.2.5.3 The Wide and Deep Model (WDM)

This model represents a novel fusion of shallow and deep learning methodologies, offering a holistic solution to classification tasks like phishing detection [1]. Designed using the Functional API (not shown in the figure), the WDM architecture incorporates both wide and deep components to capitalize on their complementary strengths (refer to Figure 3.7). Wide models excel at memorizing intricate details from data, while deep models uncover complex patterns. The WDM leverages these strengths for



image

Figure 3.6: Architecture of the Deep Neural Network (DNN) model.

robust classification.

The wide component, characterized by a series of densely connected layers, serves as the memorization engine, adept at capturing sparse, high-dimensional features from the input data. In contrast, the deep component, featuring multiple stacked layers, excels at learning hierarchical data representations and discerning intricate patterns and relationships.

Within the wide model, two dense layers with ReLU activation functions facilitate efficient feature extraction and transformation. The deep model, on the other hand, consists of four dense layers leveraging the same ReLU activation functions. To seamlessly integrate these components, the Concatenate layer merges their outputs, enabling information flow between the branches. The amalgamated model results in a final layer comprising a lone neuron, utilizing a sigmoidal activation function to produce a probability score for phishing attempts.

Training of the WDM is orchestrated through the Adam optimizer, a robust gradient-based optimization algorithm. Binary crossentropy, the preferred loss function, quantifies the discrepancy between predicted probabilities and true class labels during train-

ing. Evaluation hinges on accuracy, a fundamental metric assessing the model’s efficacy. By meticulously fine-tuning hyperparameters and optimizing architectural nuances, the goal is to craft a phishing detection system that is both robust and reliable, leveraging the synergistic capabilities of the WDM.

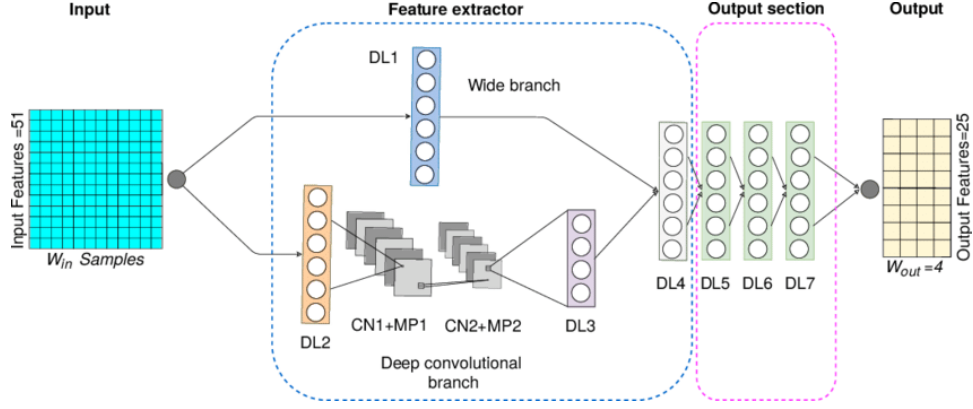


Figure 3.7: Architecture of the Wide and Deep model.

3.2.5.4 The TabNet Model

This model features an innovative deep learning framework tailored for analyzing tabular data [1] and demonstrates proficiency across diverse classification tasks, including phishing detection (as depicted in Figure 3.8). Unlike conventional models, this approach utilizes sequential attention mechanisms to prioritize informative features within the dataset, thereby enhancing its classification prowess.

The TabNet model instantiation begins with a pretraining phase using a TabNet Pretrainer on the standardized input data. This provides the model with a fundamental grasp of the data distribution, aiding in more efficient learning in subsequent training phases. The TabNet model for classification is then constructed, with key hyperparameters (refer to Figure 3.8) meticulously configured to balance model complexity and generalization capacity.

Training unfolds through epochs, with early stopping to prevent overfitting and expedite convergence. The Adam optimizer, a robust gradient-based optimization algorithm, is employed. During the training process, the model’s effectiveness is assessed

Chapter 4

Methodology

The methodology for enhancing phishing attack detection via machine learning in cybersecurity follows a systematic approach. A key challenge lies in the availability of comprehensive datasets due to the dynamic nature of phishing websites. To address this, researchers utilize features extracted from PhishTank, a repository for phishing data. This facilitates standardized performance comparisons across models. Figure 4.1 illustrates the systematic methodology adopted in this research, providing a visual roadmap for understanding the approach.

4.0.1 Data Collection

The first stage of the experimentation aimed to gather varied datasets for training and assessing machine learning models designed for phishing detection. A critical aspect of this process involved understanding the fundamental structure of URLs, the primary identifiers used to access web resources. Figure 4.2 illustrates the five individual components of a URL: Directory, Parameters, Domain, File, and the entire URL structure.

The importance of understanding URL structure lies in its role as a feature extraction technique. Researchers who compiled the Mendeley 2020 dataset leveraged this knowledge by parsing URLs into their constituent components (as shown in Figure 4.2). This parsing process enabled the extraction of 17 distinct features, each corresponding to the count of specific characters within each URL component (e.g., number of periods

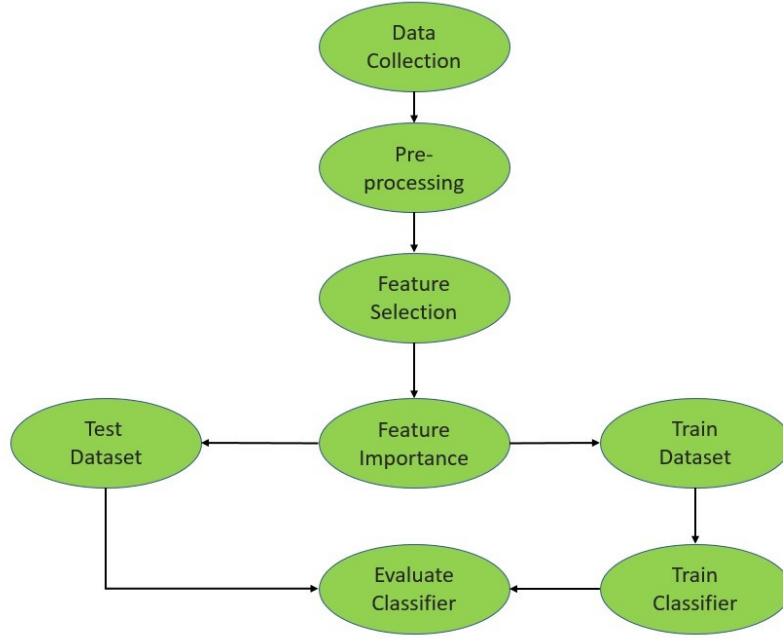


Figure 4.1: Flow diagram illustrating the proposed classification system for fall detection based on machine learning.

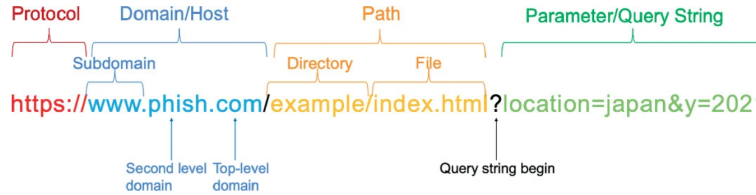


Figure 4.2: An Example of URL Structure.

in the domain, number of forward slashes in the directory, etc.). Extracting these features from a diverse set of URLs provided a rich source of information for subsequent model training and analysis.

The meticulous data collection approach extended beyond the Mendeley 2020 dataset. Diverse datasets that encapsulated a broad spectrum of phishing tactics were actively sought, ensuring the models could adapt to various real-world scenarios. The overarching goal of this data collection effort was to establish a robust foundation for training machine learning models that could effectively address the challenges of phishing detection in diverse contexts.

4.0.2 Preprocessing

The preprocessing stage is integral to ensuring the quality and uniformity of the dataset. Key tasks in this phase encompass addressing single-valued columns, eliminating instances with zero or negative values, and imputing missing values with the mean. Additionally, the application of Min-Max normalization transforms the data, ensuring uniform scaling across features and further enhancing dataset integrity. Collectively, these preprocessing steps contribute to the completeness and reliability of the dataset, laying the groundwork for subsequent analyses and model training.

4.0.3 Feature Selection

Selecting features is a critical stage in model development, greatly impacting the overall effectiveness of machine learning models. In this project, a rigorous feature selection approach was employed, primarily through comprehensive correlation analysis. The objective is to identify features with a correlation coefficient exceeding 0.8, highlighting strong linear relationships between variables. The correlation analysis serves as a foundational step, strategically laying the groundwork for subsequent model training and testing phases. By pinpointing highly correlated features, the focus is on streamlining the dataset, focusing on those variables that contribute most significantly to the desired outcomes. This meticulous feature selection process is essential for optimizing model performance and ensuring that the selected features collectively contribute to the model's ability to make accurate predictions. Through this approach, the efficiency of the dataset is enhanced, facilitating improved model training and, ultimately, more robust and accurate machine learning models for phishing attack detection.

4.0.4 Feature Importance

Recognizing the importance of features is crucial for building effective machine learning models. Permutation importance analysis played a vital role in assessing the significance of each feature across different models, including the Feedforward Neural Net-

work (FNN), TabNet, Wide and Deep, and Deep Neural Network (DNN). This analysis involves systematically shuffling feature values to observe their impact on model performance. Features causing a notable decline in accuracy when permuted are deemed crucial, indicating their substantial influence on model predictions. After completing permutation importance analysis for all models, the resulting scores are averaged and ranked, highlighting the most influential features. This prioritization process is essential for refining model architecture and enhancing predictive capabilities. By employing permutation importance analysis, valuable insights are obtained, contributing towards crafting resilient and precise machine learning models for detecting phishing attacks.

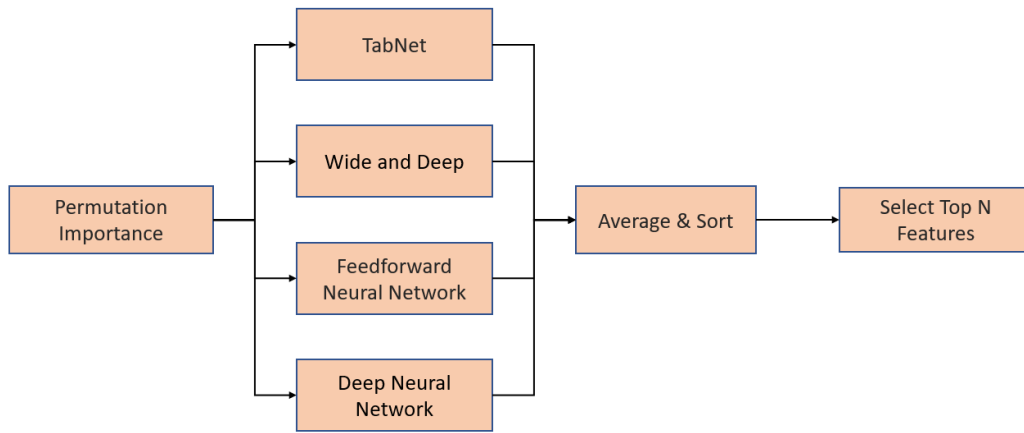


Figure 4.3: Flowchart illustrating the Permutation Importance-based Feature Selection process in the machine learning pipeline.

4.0.5 Train and Test Model

The model development phase is advancing with the implementation, training, and testing of the Feedforward Neural Network (FNN), TabNet, Wide and Deep, and Deep Neural Network (DNN) models. This comprehensive process involves training each model individually using the top N features meticulously identified through the feature selection approach. To further enhance the reliability of the findings, a rigorous 10-fold cross-validation methodology will be applied, ensuring a thorough evaluation of each model's performance. This iterative process not only refines the models but also

establishes a robust foundation for subsequent analyses and optimizations.

4.0.6 Evaluation

The final stage, evaluation, is dedicated to assessing the developed models using key metrics such as true positive rate (TPR), accuracy, false positive rate (FPR), and testing time. These metrics offer valuable insights into how effectively the machine learning models detect phishing attacks. The evaluation process involves both quantitative analysis and qualitative assessments to comprehensively evaluate the models' effectiveness. The accuracy metric reflects how accurately the model makes predictions, while TPR and FPR indicate its ability to accurately detect phishing instances while minimizing false alarms. Testing time indicates the computational efficiency of the models. Subsequently, the 'Anti-Phishing Score' is computed based on these metrics, providing a holistic measure that considers various factors crucial for effective phishing attack detection. This thorough evaluation is integral to validating the practical utility of the machine learning models in practical scenarios and guiding further refinements and optimizations to ensure their effectiveness against the dynamic nature of phishing attacks.

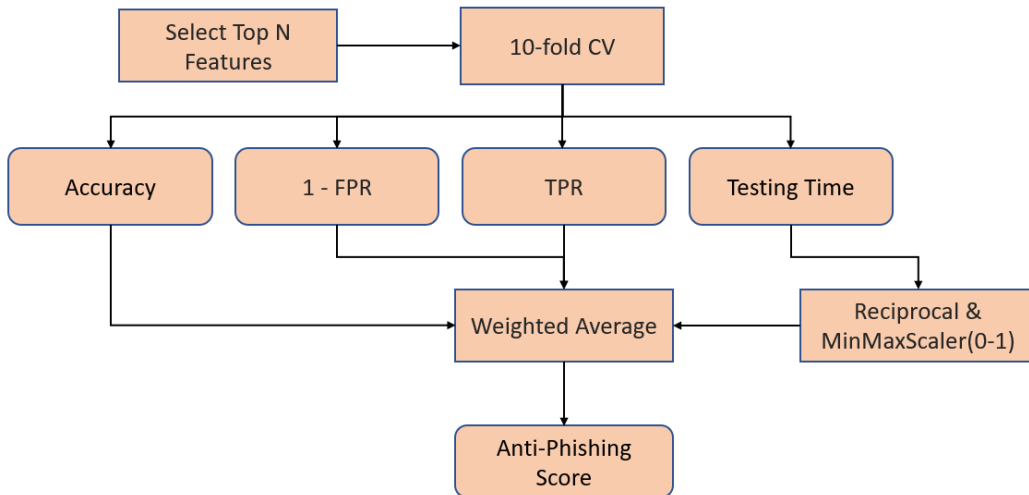


Figure 4.4: Evaluation Workflow for the Feature Selection Approach.

Chapter 5

Results

5.1 Data Cleaning and Data Transformation

5.1.1 Data Cleaning and Consistency

In the initial phase of the project, the focus was on refining the dataset through robust data cleaning strategies. The following key steps were implemented:

1. **Removal of Uninformative Features:** Features with a single unique value were identified and excluded, streamlining the dataset and eliminating redundancy.
2. **Handling of Aberrant Values:** Instances with values of 0 or -1 were identified as outliers and removed to enhance the overall quality of the data.
3. **Imputation of Missing Values:** Missing values were addressed by employing the mean imputation technique, ensuring a comprehensive and complete dataset.

These meticulous data cleaning efforts not only eliminated noise and inconsistencies but also contributed to the reduction of features from an initial count of 111 to a more manageable 91. This foundational work establishes a clean and consistent dataset, forming the basis for subsequent stages of feature selection and advanced model development.

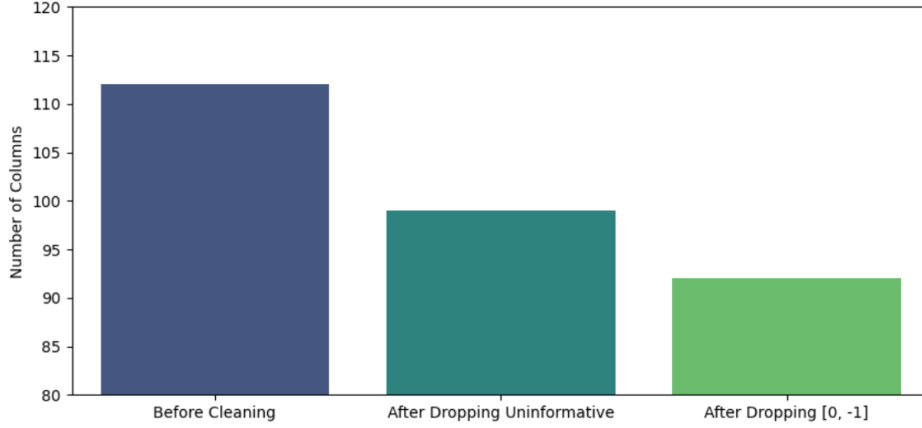


Figure 5.1: Number of Columns Before and After Data Cleaning.

5.1.2 Data Transformation: Min-Max Normalization

As part of the data transformation phase, Min-Max normalization was applied to standardize the range of numerical features. This technique scales the values of each feature to a specific range, typically $[0, 1]$, preserving the relationships between data points and preventing certain features from dominating the model due to their larger scale.

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

Here:

- X stands for the starting value of the feature.
- X_{\min} refers to the smallest feature value within the dataset.
- X_{\max} indicates the largest feature value within the dataset.

This transformation ensures that all features contribute equally to the model training process, preventing bias towards variables with larger numerical ranges. The completed Min-Max normalization sets the stage for improved model convergence and performance during subsequent training and evaluation phases.

5.2 Feature Selection

5.2.1 Correlation-based Feature Selection

In order to enhance the dataset's efficiency and facilitate optimal model training, an intricate feature selection process was undertaken. Correlation analysis identified features with a correlation coefficient exceeding 0.8, capturing highly correlated features. The correlation coefficient measures the extent of the linear association between two variables, with values ranging from -1 to 1. A coefficient near 1 signifies a strong positive correlation, while a coefficient near -1 indicates a strong negative correlation. Features with high correlation were carefully assessed for their impact on the overall dataset.

The formula for calculating the Pearson correlation coefficient (r) between two variables X and Y is expressed as:

$$r = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}}$$

In this equation, X_i and Y_i represent individual data points within variables X and Y , respectively, while \bar{X} and \bar{Y} denote their respective means.

This meticulous feature selection addresses multicollinearity and lays the groundwork for creating a refined subset of features, optimizing the dataset for subsequent model training and validation.

5.2.2 Feature Importance

Understanding feature importance is essential for developing effective phishing detection models. Permutation importance analysis offers valuable insights by systematically evaluating the impact of each feature on model predictions. This method entails rearranging feature values and assessing the subsequent impact on model accuracy.

Features that exhibit a notable decline in accuracy after rearrangement are considered significant, informing future choices regarding feature prioritization and dataset refinement.

In the pursuit of advancing phishing detection capabilities, permutation importance analysis emerged as a foundational element of the methodology. By applying this rigorous approach across four distinct models — Proposed Neural Network (PNN), Feedforward Neural Network (FNN), TabNet, and Wide and Deep model — the top 20 features crucial in distinguishing phishing websites were identified. Notably, the analysis revealed commonalities across models, highlighting features such as 'time_domain_activation' and 'qty_slash_url' as robust discriminators, consistently influential across different architectures.

Additionally, permutation importance analysis provided insights specific to each model architecture, illustrating feature importance rankings visually. Figures 5.3, 5.2, 5.5, and 5.4 depict the feature importance for the PNN, FNN, TabNet, and Wide and Deep models, respectively. These visualizations complement the tabular representation of top features, enhancing the understanding of critical factors driving model performance.

These discoveries greatly enhance model interpretability and assist in pinpointing vital features necessary for precise phishing detection. Through employing permutation importance analysis across diverse model architectures, it guarantees the creation of more efficient and durable phishing detection systems, crucial for tackling the dynamic nature of phishing attacks. Furthermore, the feature importance rankings obtained for each model are averaged to create a comprehensive list. This consolidated list serves as a basis for model training and evaluation in the subsequent sections, facilitating a unified approach towards enhancing phishing detection capabilities.

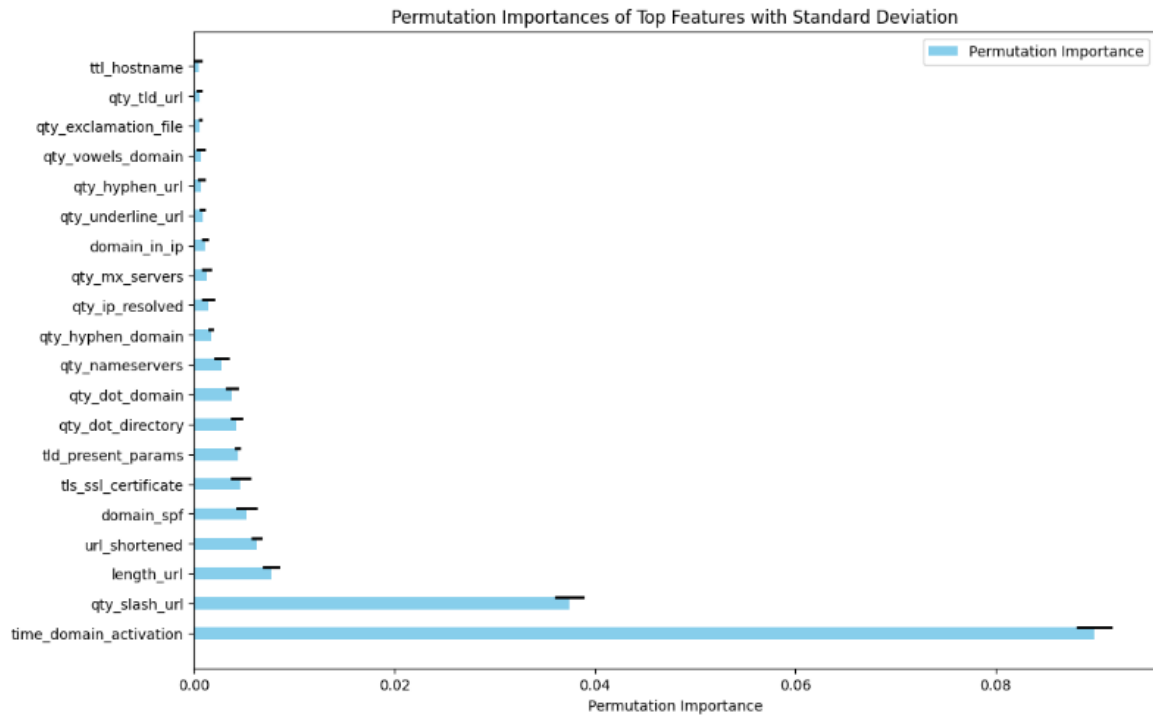


Figure 5.2: The Permutation Importance Bar chart and Ranking depict the significance of features in the Feedforward Neural Network Model.

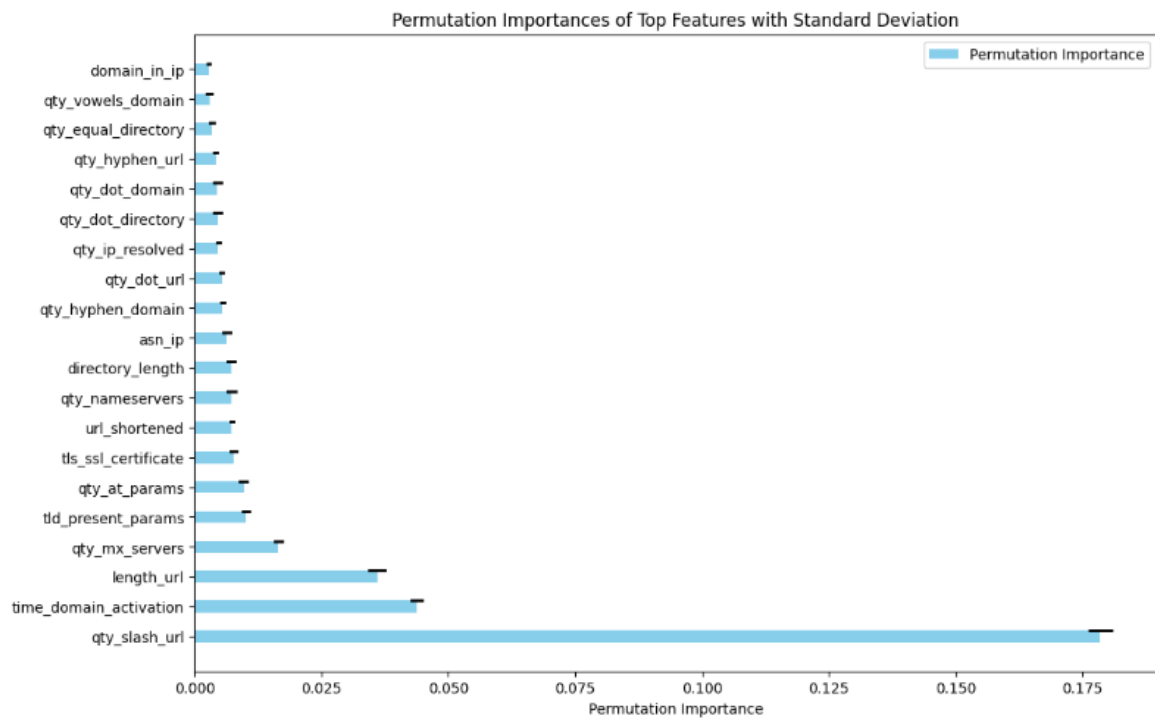


Figure 5.3: The Permutation Importance Bar chart and Ranking depict the significance of features in the Deep Neural Network Model.

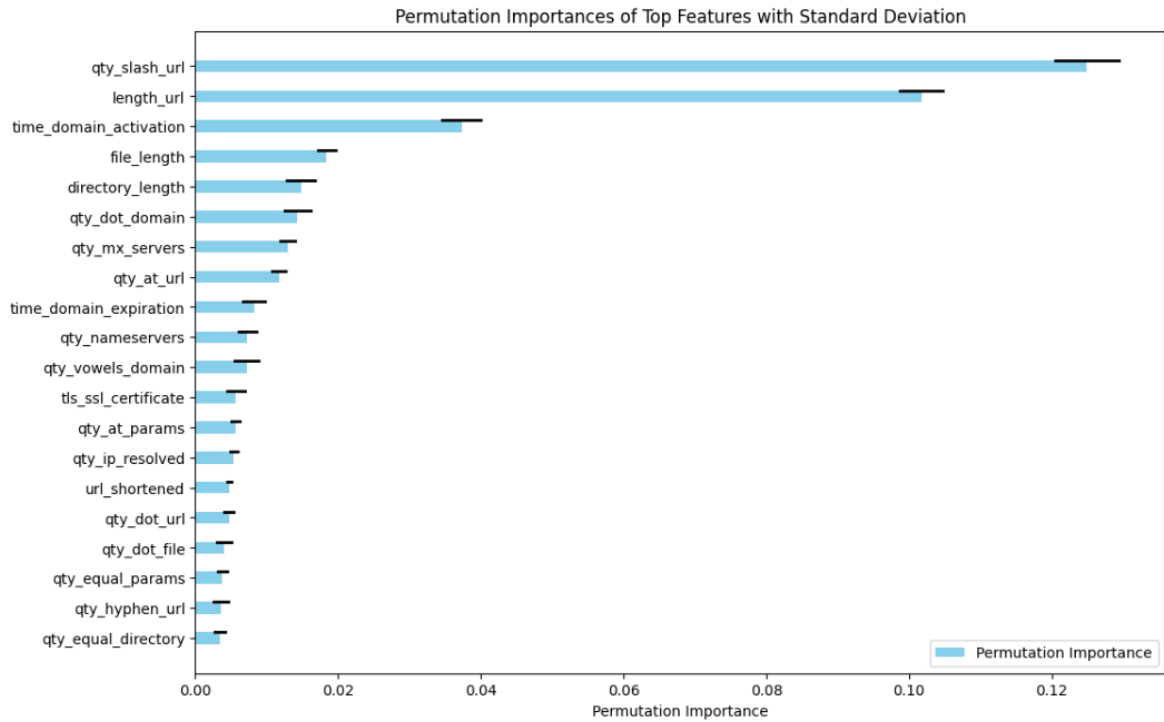


Figure 5.4: The Permutation Importance Bar chart and Ranking depict the significance of features in the Wide and Deep Model.

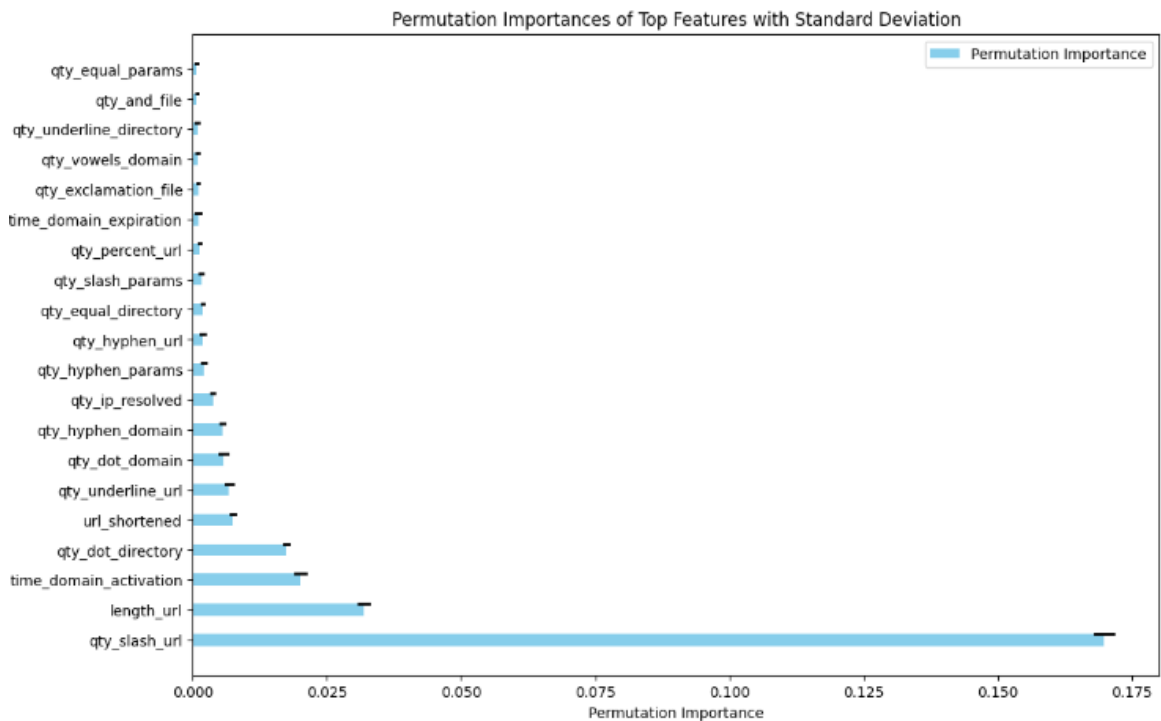


Figure 5.5: The Permutation Importance Bar chart and Ranking depict the significance of features in the TabNet Model.

Table 5.1: Top Features Selected through Permutation Importance Analysis

| Feedforward Neural Network | Deep Neural Network | TabNet | Wide and Deep |
|-----------------------------------|----------------------------|-------------------------|------------------------|
| time_domain_activation | qty_slash_url | qty_slash_url | qty_slash_url |
| qty_slash_url | time_domain_activation | length_url | length_url |
| length_url | length_url | time_domain_activation | time_domain_activation |
| url_shortened | qty_mx_servers | qty_dot_directory | file_length |
| domain_spf | tld_present_params | url_shortened | directory_length |
| tls_ssl_certificate | qty_at_params | qty_underline_url | qty_dot_domain |
| tld_present_params | tls_ssl_certificate | qty_dot_domain | qty_mx_servers |
| qty_dot_directory | url_shortened | qty_hyphen_domain | qty_at_url |
| qty_dot_domain | qty_nameservers | qty_ip_resolved | time_domain_expiration |
| qty_nameservers | directory_length | qty_hyphen_params | qty_nameservers |
| qty_hyphen_domain | asn_ip | qty_hyphen_url | qty_vowels_domain |
| qty_ip_resolved | qty_hyphen_domain | qty_equal_directory | tls_ssl_certificate |
| qty_mx_servers | qty_dot_url | qty_slash_params | qty_at_params |
| domain_in_ip | qty_ip_resolved | qty_percent_url | qty_ip_resolved |
| qty_underline_url | qty_dot_directory | time_domain_expiration | url_shortened |
| qty_hyphen_url | qty_dot_domain | qty_exclamation_file | qty_dot_url |
| qty_vowels_domain | qty_hyphen_url | qty_vowels_domain | qty_dot_file |
| qty_exclamation_file | qty_equal_directory | qty_underline_directory | qty_equal_params |
| qty_tld_url | qty_vowels_domain | qty_and_file | qty_hyphen_url |
| ttl_hostname | domain_in_ip | qty_equal_params | qty_equal_directory |

Table 5.2: Top Features Selected through Permutation Importance Analysis

| Feature | Description |
|------------------------|-----------------------------------|
| qty_slash_url | Quantity of slashes in URL |
| time_domain_activation | Time since domain activation |
| length_url | Length of URL |
| qty_mx_servers | Quantity of MX servers |
| qty_dot_directory | Quantity of dots in directory |
| qty_dot_domain | Quantity of dots in domain |
| url_shortened | Presence of URL shortening |
| directory_length | Length of directory |
| file_length | Length of file |
| tls_ssl_certificate | Presence of TLS/SSL certificate |
| qty_nameservers | Quantity of nameservers |
| qty_at_params | Quantity of '@' in parameters |
| qty_ip_resolved | Quantity of resolved IP addresses |
| tld_present_params | TLD presence in parameters |
| qty_hyphen_domain | Quantity of hyphens in domain |
| qty_at_url | Quantity of '@' in URL |
| qty_vowels_domain | Quantity of vowels in domain |
| qty_hyphen_url | Quantity of hyphens in URL |
| time_domain_expiration | Time until domain expiration |
| domain_spf | Presence of SPF in domain |

5.3 Model Implementation

The project initiates with the setup of four key models, namely FNN (Feedforward Neural Network), DNN (Deep Neural Network), Wide and Deep, and TabNet, for the specific purpose of permutation importance analysis. These models serve as the base for understanding the significance of individual features and their impact on model performance. Additionally, they form the foundation for subsequent model training and evaluation in the upcoming sections. It will be used for feature importance using permutation importance.

Each model’s architecture is tailored to address the complexities of phishing detection while leveraging the strengths of different neural network architectures. Appendix A presents the code implementations for the FNN, DNN, Wide and Deep, and TabNet models used in phishing detection. These architectures are designed to extract meaningful features from the input data, capture intricate patterns indicative of phishing attempts, and provide robust predictions. By implementing these models, the groundwork for comprehensive analysis and evaluation is laid, aiming to develop effective phishing detection systems.

5.4 Model Training and Evaluation

5.4.1 Model Training

The model training process commenced with the setup of four key models: Wide and Deep, Deep Neural Network (DNN), TabNet, and Feedforward Neural Network (FNN). These models served as the foundation for the investigation into feature importance using permutation importance analysis. The training phase involved ten-fold cross-validation to ensure robustness and generalizability across different datasets. During each fold of cross-validation, various metrics, including accuracy, false positive rates (FPR), testing time, and true positive rates (TPR), were meticulously monitored to capture the performance dynamics of each model comprehensively.

Figures 5.6, 5.7, 5.8 and 5.9 showcase the performance metrics of the four models across ten folds of cross-validation. The plots illustrate the variations in accuracy, testing time, FPR, and TPR throughout the cross-validation process. By averaging these metrics across all folds, a holistic view of each model’s performance is obtained, enabling the identification of trends and the assessment of stability across different datasets.

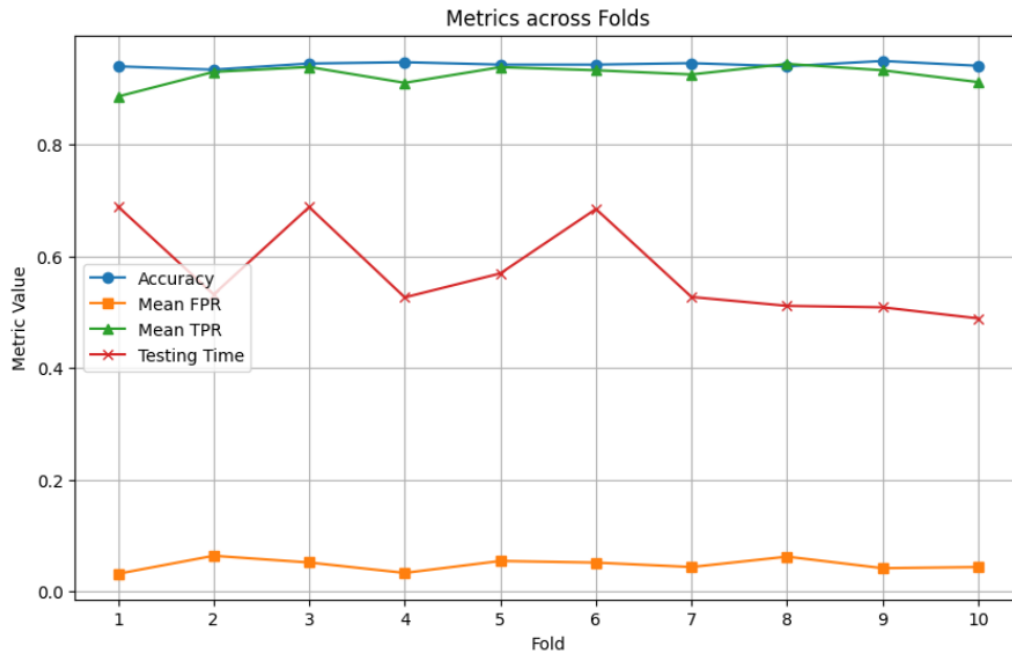


Figure 5.6: Feedforward Neural Network Model Evaluation Results Across Ten-Fold Cross Validation.

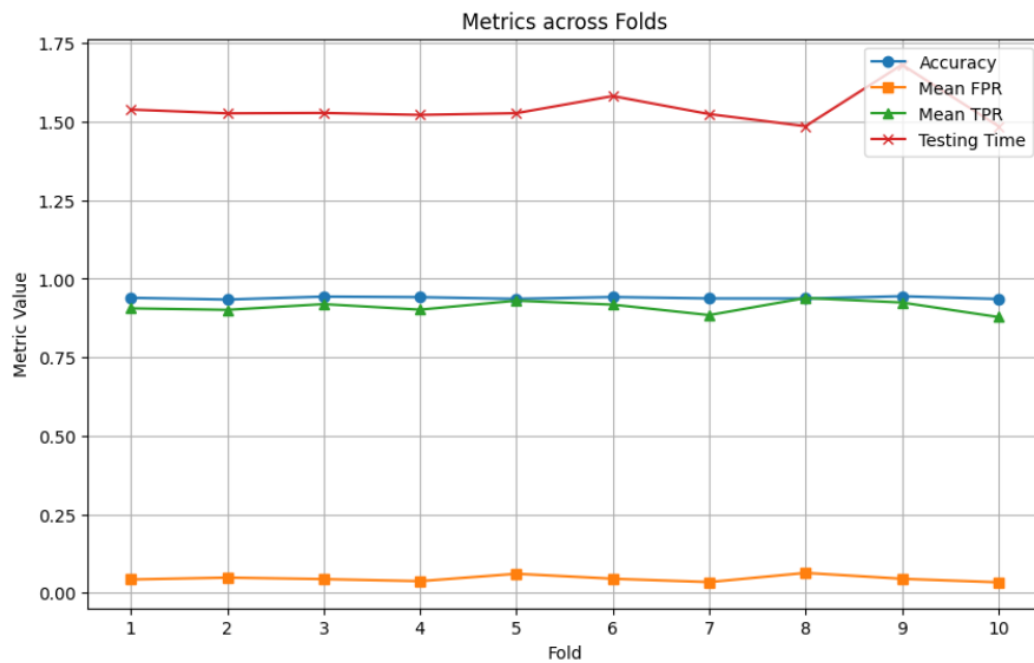


Figure 5.7: Deep Neural Network Model Evaluation Results Across Ten-Fold Cross Validation.

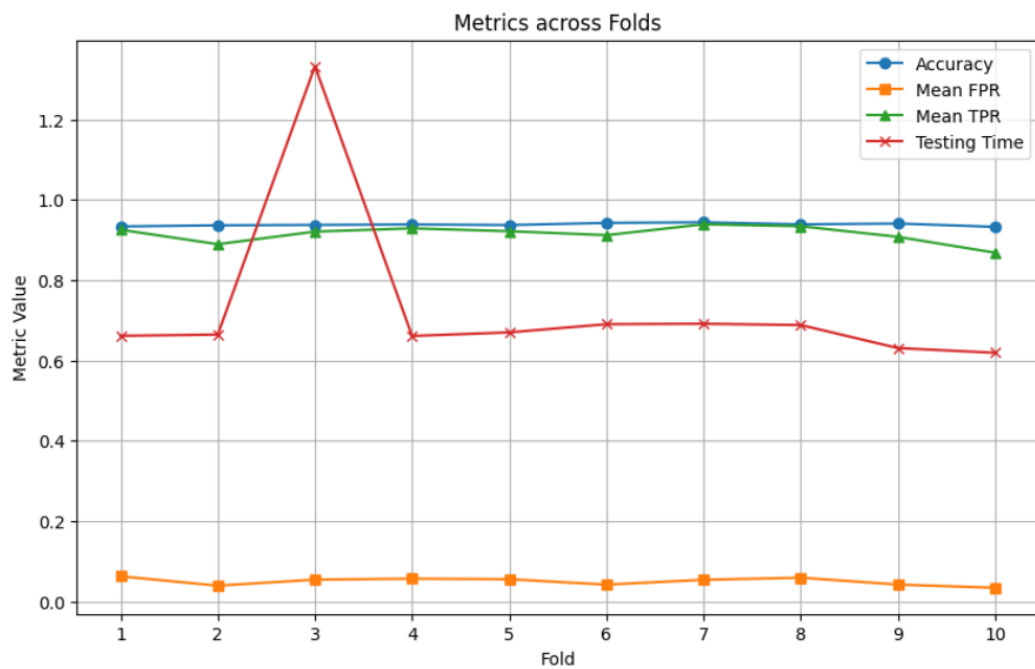


Figure 5.8: Wide & Deep Model Evaluation Results Across Ten-Fold Cross Validation.

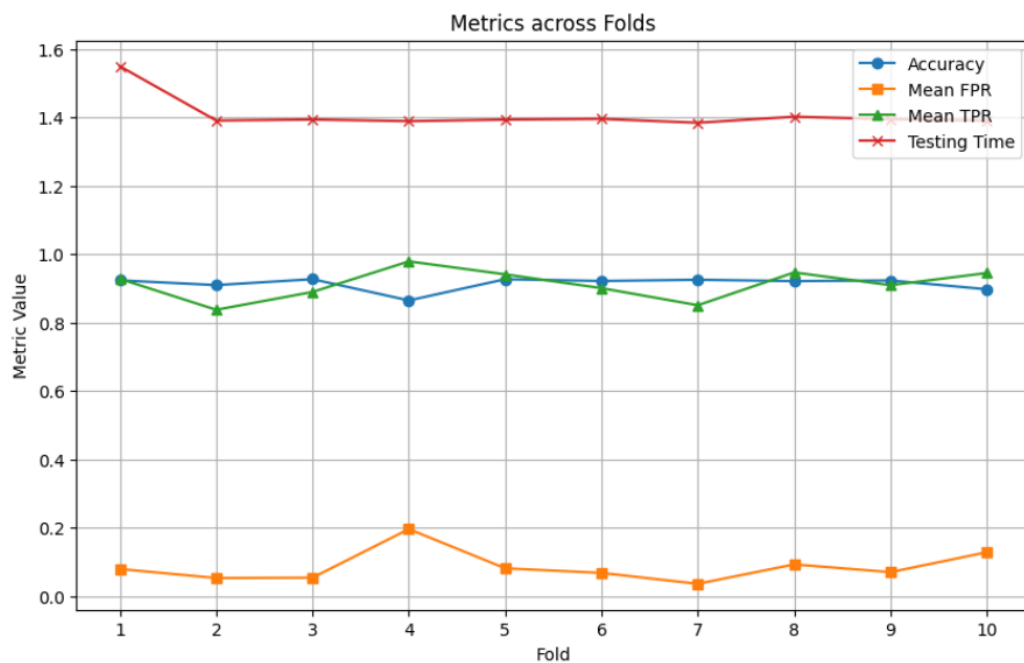


Figure 5.9: TabNet's Evaluation Results Across Ten-Fold Cross Validation.

5.4.2 Model Evaluation

The model evaluation process underwent thorough scrutiny, utilizing ten-fold cross-validation to offer a comprehensive evaluation of each model’s performance in phishing detection. Diverse metrics such as false positive rates, testing time, true positive rates, and accuracy were explored to understand the effectiveness of the models. However, the evaluation didn’t stop there. The anti-phishing score, a weighted amalgamation of performance metrics, was introduced to obtain a holistic view of model effectiveness. This holistic approach considers not only traditional metrics like accuracy, FPR, and TPR but also incorporates testing time, crucial in real-world applications where efficiency is paramount.

The anti-phishing score calculation formula is as follows:

$$\text{Anti-Phishing Score} = 0.3 \times \text{Accuracy} + 0.25 \times (1 - \text{FPR}) + 0.25 \times \text{TPR} + 0.2 \times \text{Testing Time}$$

Each component of the anti-phishing score formula contributes uniquely to the overall assessment. Accuracy reflects the model’s proficiency in correctly classifying instances, while FPR measures the proportion of legitimate instances misclassified as phishing. TPR represents the ratio of genuine phishing instances accurately recognized as such, highlighting the model’s effectiveness in identifying phishing attacks. Testing time reflects operational efficiency, crucial in real-world scenarios. By incorporating scaled testing time, the anti-phishing score accounts for computational efficiency, ensuring that longer processing times are appropriately weighted to reflect both accuracy and operational efficiency.

Table 5.3: Model Performance Metrics

| Model | Accuracy | FPR | TPR | Testing Time | Anti-Phishing Score |
|-------------|----------|--------|--------|--------------|---------------------|
| DNN | 0.9392 | 0.0456 | 0.9103 | 0.6492 | 0.7479 |
| Feedforward | 0.9427 | 0.0479 | 0.9250 | 1.7470 | 0.9521 |
| TabNet | 0.9382 | 0.0494 | 0.9147 | 1.3685 | 0.7420 |
| Wide & Deep | 0.9139 | 0.0854 | 0.9126 | 0.7100 | 0.8788 |

The visualization of model performance metrics through ROC and precision-recall curves provides invaluable insights into the efficacy of phishing detection models. Figures 5.10 and 5.11 illustrate ROC and precision-recall curves, respectively, for the four different models: Feedforward, DNN, Wide & Deep, and TabNet, along with a random classifier for comparison. ROC curves demonstrate the balance between true positive rate and false positive rate, whereas precision-recall curves illustrate the equilibrium between precision and recall. These graphical representations aid in making informed decisions by providing insights into how each model manages the balance between true positives and false positives, as well as precision and recall, which are vital factors in real-world phishing detection scenarios. Among the diverse range of models explored, one emerged as the clear frontrunner—the Feedforward Neural Network (FNN). With an impressive anti-phishing score of 0.9521, it ascended as the pinnacle of efforts in phishing detection.

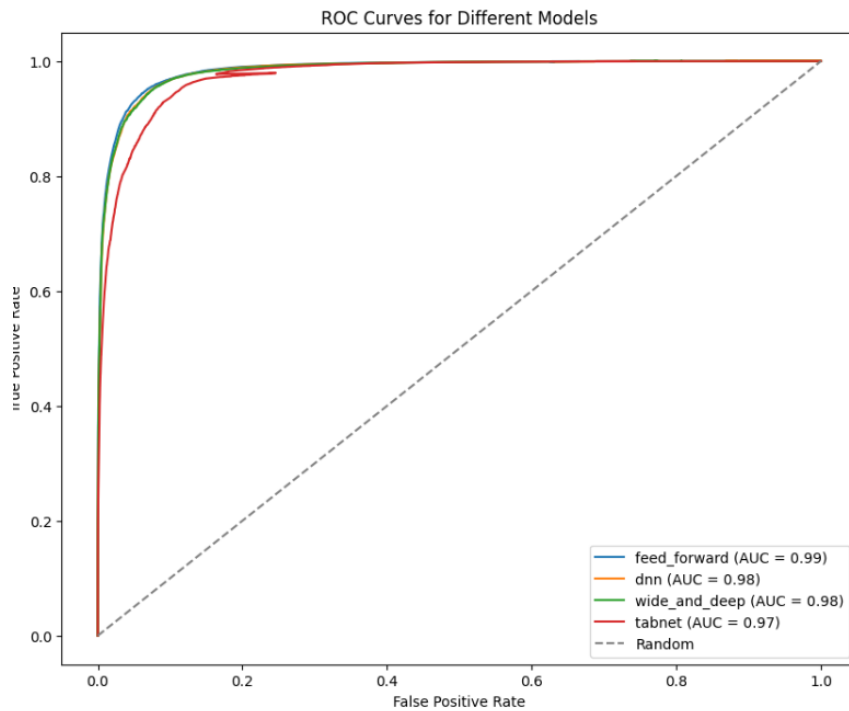


Figure 5.10: ROC Curves

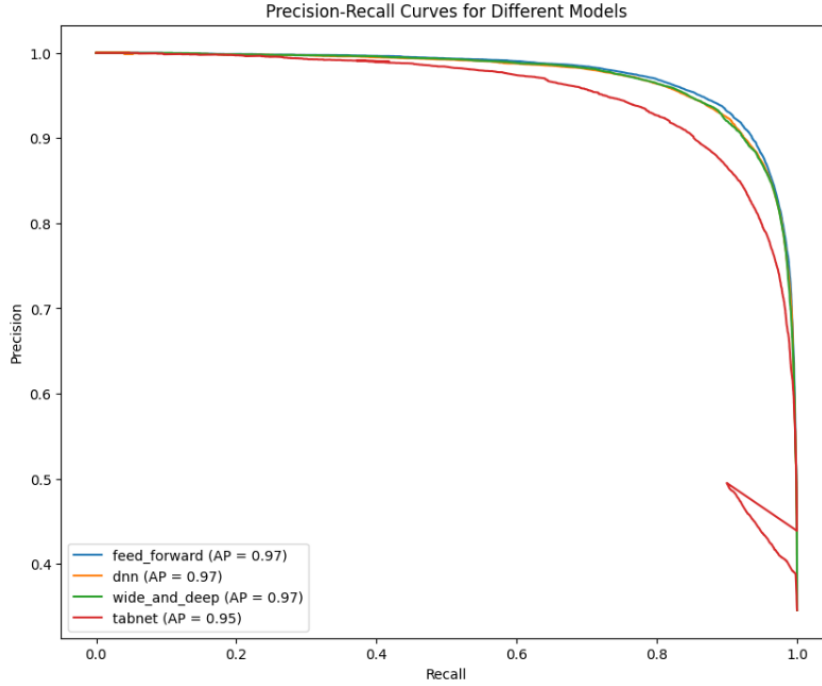


Figure 5.11: Precision-Recall Curves

5.5 Optimizing Model Performance: Grid Search Hyperparameter Tuning

The objective of the hyperparameter grid search was to find a middle ground between model effectiveness and computational efficiency, ensuring proficient phishing detection without sacrificing accuracy. Table 5.4 presents the results of this search, showcasing the best-performing configuration alongside an alternative model with slightly different parameters.

The best result, achieved with 20 columns of interest and trained over 40 epochs, demonstrated a marginally higher accuracy of 95.53% on the training data compared to the considered model, which utilized 14 columns of interest and underwent 50 epochs, resulting in an accuracy of 94.46%. However, despite the small difference in accuracy, the considered model offers advantages in terms of computational efficiency. It achieves equilibrium between precision and computational resources by decreasing the selected

columns and marginally boosting the epoch count, thereby cutting down on computation time significantly while upholding a notable accuracy level, rendering it apt for real-time phishing detection applications where efficiency is crucial.

Furthermore, both models utilize a test size of 0.2 for robust validation and evaluation, ensuring reliable performance assessment. The choice of learning rate, batch size, and optimizer remains consistent across both configurations, emphasizing their importance in model training and convergence. Upon evaluation on new data, the considered model achieved an accuracy of 80%, demonstrating its generalizability and effectiveness in real-world scenarios. Despite the slight decrease in accuracy compared to the best result, this model offers practical advantages in terms of computational efficiency, making it a preferred choice for implementation in real-time defenses against phishing.

As mentioned in the evaluation process, the new dataset utilized is from 2021, as specified in Section 3.1, indicating its contemporaneity. Consequently, achieving an accuracy rate of 80% with this dataset is noteworthy, given the constantly evolving landscape of phishing strategies and the associated detection challenges.

Table 5.4: Hyperparameter Grid Search Results

| Parameters | Best Result | Considered Model |
|----------------------|--------------------|-------------------------|
| Columns of Interest | 20 | 14 |
| Epochs | 40 | 50 |
| Test Size | 0.2 | 0.2 |
| Batch Size | 128 | 64 |
| First Hidden Layer | 64 | 128 |
| Learning Rate | 0.01 | 0.001 |
| Second Hidden Layer | 64 | 64 |
| Optimizer | Adam | Adam |
| Accuracy on Data | 0.9553 | 0.9446 |
| Accuracy on New Data | 81.05% | 80% |

5.6 Automated Script

In the quest to fortify phishing detection methodologies, a groundbreaking amalgamation of feature selection and hyperparameter tuning scripts is introduced. This innovative integration signals a pivotal advancement, propelling us into an era of automated vigilance against evolving phishing threats. Empowered by this sophisticated script, one is poised to navigate the dynamic landscape of cyber threats with unprecedented efficacy and foresight. This automated script signifies the culmination of thorough research and development endeavors, utilizing advanced techniques in data science and the latest advancements in ML. By seamlessly combining feature selection algorithms and hyperparameter tuning strategies, the script streamlines the model development process, maximizing both performance and efficiency.

With this script at one's disposal, traditional manual approaches to model optimization are transcended, harnessing the power of automation to accelerate innovation and stay ahead of emerging threats. Its adaptability and scalability ensure that detection systems remain agile and responsive in the face of evolving attack vectors and malicious tactics. Furthermore, the modularity and extensibility of the script enable seamless integration with existing infrastructure and workflows, fostering collaboration and knowledge sharing within the cybersecurity ecosystem. By democratizing access to advanced modeling techniques, stakeholders across disciplines are empowered to contribute to the collective defense against phishing attacks.

Chapter 6

Conclusion and Future Scope

6.1 Work Summary

The study addressed the development of effective phishing detection methodologies leveraging machine learning techniques. The objective was to identify robust models for accurately detecting phishing attempts to enhance cybersecurity measures.

To achieve this objective, a methodology focused on evaluating various machine learning models, including TabNet, Deep Neural Network (DNN), Wide and Deep, and Feedforward Neural Network (FNN), was adopted. Feature importance analysis using permutation importance techniques was conducted to identify key features crucial for distinguishing phishing websites. Additionally, a ten-fold cross-validation approach was implemented to comprehensively evaluate each model's performance, considering metrics such as false positive rates (FPR), testing time, accuracy, and true positive rates (TPR).

6.2 Conclusions

In addition to achieving exceptional accuracy, the study emphasizes the broader implications of the findings in the cybersecurity landscape. The identification of the Feedforward Neural Network (FNN) as the most effective model underscores the critical role of advanced machine learning techniques in fortifying cybersecurity defenses. By

leveraging deep learning algorithms, organizations and individuals can better defend against the ever-evolving tactics employed by malicious actors in phishing attacks.

Furthermore, meticulous execution of hyperparameter tuning not only resulted in high accuracy but also highlighted the importance of fine-tuning model parameters for optimal performance. The remarkable accuracy rate achieved through this approach serves as a testament to its effectiveness in enhancing model efficacy. This underscores the necessity for continuous refinement and optimization of machine learning models to adapt to changing threat landscapes and ensure robust protection against cyber threats. Additionally, the identification of carefully selected features further enhances the efficiency of the model in distinguishing phishing attempts from legitimate web traffic. By leveraging these key features, the model demonstrates not only accuracy but also efficiency, enabling swift and reliable detection of phishing attacks. This efficiency is crucial in real-world scenarios where rapid identification and response are essential for mitigating potential risks and minimizing the impact of cyber threats.

Overall, the study underscores the importance of rigorous experimentation, evaluation, and optimization in developing effective phishing detection models. By integrating advanced machine learning techniques with meticulous tuning and feature selection, the groundwork is laid for more resilient cybersecurity frameworks capable of defending against sophisticated phishing attacks. Through continued research and innovation in this field, the state-of-the-art in cybersecurity can be advanced, empowering organizations and individuals to stay ahead of emerging threats.

6.3 Future Scope of Work

In the future, numerous opportunities for further research and advancement in phishing detection are available. Future work could involve exploring deep learning optimization methods to further enhance model performance. This includes investigating novel architectures, fine-tuning hyperparameters, and leveraging larger datasets for training and evaluation. Additionally, there is scope for refining and optimizing automated

scripts for phishing detection, integrating advanced features such as real-time URL analysis and behavior-based detection techniques. Furthermore, deploying phishing detection frameworks in practical settings, such as web browsers and email clients, holds promise for providing users with real-time protection against phishing threats. Overall, the future scope of work entails ongoing enhancements and progression of phishing detection methodologies, utilizing state-of-the-art machine learning techniques to proactively address evolving cyber threats and uphold resilient cybersecurity measures.

Appendices

Appendix A

Code

A.1 Architecture of Feedforward Neural Network (FNN) Model

```
def create_feedforward_neural_network():  
    # Create a sequential model  
    model = Sequential()  
  
    # Add input layer with 128 neurons and ReLU activation  
    model.add(Dense(128, input_dim=X_train.shape[1], activation='relu'))  
  
    # Add hidden layer with 64 neurons and ReLU activation  
    model.add(Dense(64, activation='relu'))  
  
    # Add output layer with sigmoid activation  
    model.add(Dense(1, activation='sigmoid'))  
  
    # Compile the model with Adam optimizer and binary crossentropy loss  
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])  
  
    return model
```

A.2 Architecture of DNN (Deep Neural Network) Model

```
def deep_neural_network_model():  
    # Create a sequential model  
    model = Sequential()  
  
    # Add input layer with 64 neurons and ReLU activation
```

```

model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],)))
# Add hidden layer with 64 neurons and ReLU activation
model.add(Dense(64, activation='relu'))
# Add output layer with sigmoid activation
model.add(Dense(1, activation='sigmoid'))
# Compile the model with Adam optimizer and binary crossentropy loss
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
return model

```

A.3 Architecture of Wide & Deep Model

```

def build_wide_deep_model(input_shape):
    # Define the wide and deep model using Functional API
    wide_input = Input(shape=(input_shape,))
    deep_input = Input(shape=(input_shape,))
    wide_model = Dense(32, activation='relu')(wide_input)
    wide_model = Dense(16, activation='relu')(wide_model)
    deep_model = Dense(64, activation='relu')(deep_input)
    deep_model = Dense(32, activation='relu')(deep_model)
    combined_model = Concatenate()([wide_model, deep_model])
    output_layer = Dense(1, activation='sigmoid')(combined_model)
    model = Model(inputs=[wide_input, deep_input], outputs=output_layer)
    # Compile the model with Adam optimizer and binary crossentropy loss
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    return model

```

A.4 Architecture of TabNet Classifier Model

```

# Create TabNet Pretrainer for pretraining
pretrainer = TabNetPretrainer()
pretrainer.fit(X_train_set)

# Create TabNet model for classification
tabnet_params = {"input_dim": X_train_set.shape[1], "output_dim": 1, "n_d": 8, "n_a": 8, "n_steps": 3}
model_tabnet = TabNetClassifier(**tabnet_params)

# Early stopping criteria
max_epochs = 100
patience = 10
best_auc = 0

```

```

counter = 0

# Training loop with early stopping
for epoch in range(max_epochs):
    model_tabnet.fit(
        X_train_set, y_train_set,
        eval_set=[(X_test_set, y_test_set)],
        eval_metric=['auc'],
        max_epochs=1,
        patience=patience, # Set patience here
        batch_size=256,
        virtual_batch_size=128,
        num_workers=0,
        drop_last=False,
        from_unsupervised=pretrainer
    )

    # Evaluate model on validation set
    preds_prob = model_tabnet.predict_proba(X_test_set)[: , 1]
    roc_auc_score = roc_auc_score(y_test_set, preds_prob)

    print(f"epoch {epoch} | val_auc: {auc_score:.5f}")

    # Early stopping
    if auc_score > best_auc:
        best_auc = auc_score
        counter = 0
        # Save the best model
        torch.save(model_tabnet, "best_model.pt")
    else:
        counter += 1
        if counter >= patience:
            print("Early stopping")
            break

# Load the top-performing model
model = torch.load("best_model.pt")

```

References

- [1] F. Salahdine, Z. E. Mrabet, and N. Kaabouch, “Phishing attacks detection: A machine learning-based approach,” in *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, New York, NY, USA, 2021, pp. 250–255.
- [2] M. Baykara and Z. Z. Gürel, “Detection of phishing attacks,” in *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, Antalya, Turkey, 2018, pp. 1–5.
- [3] K. L. Chiew, K. S. C. Yong, and C. L. Tan, “A survey of phishing attacks: Their types, vectors and technical approaches,” vol. 106, 2018, pp. 1–20.
- [4] F. Yahya *et al.*, “Detection of phishing websites using machine learning approaches,” in *2021 International Conference on Data Science and Its Applications (ICoDSA)*, Bandung, Indonesia, 2021, pp. 40–47.
- [5] A. Alswailem, B. Alabdullah, N. Alrumayh, and A. Alsedrani, “Detecting phishing websites using machine learning,” in *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*, Riyadh, Saudi Arabia, 2019, pp. 1–6.
- [6] H. Zuhair, A. Selamat, and M. Salleh, “Feature selection for phishing detection: A review of research,” vol. 15, 2016, p. 147.
- [7] P. Chinnasamy, N. Kumaresan, R. Selvaraj, S. Dhanasekaran, K. Ramprathap, and S. Boddu, “An efficient phishing attack detection using machine learning

- algorithms,” in *2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, Bhubaneswar, India, 2022, pp. 1–6.
- [8] T. R. N and R. Gupta, “Feature selection techniques and its importance in machine learning: A survey,” in *2020 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal, India, 2020, pp. 1–6.
- [9] R. Ramachandran, G. Ravichandran, and A. Raveendran, “Evaluation of dimensionality reduction techniques for big data,” in *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, 2020, pp. 226–231.
- [10] S. Dangwal and A.-N. Moldovan, “Feature selection for machine learning-based phishing websites detection,” in *2021 International Conference on Cyber Situational Awareness, Data Analytics, and Assessment (CyberSA)*, Dublin, Ireland, 2021, pp. 1–6.
- [11] M. R. Chinguwo and R. Dhanalakshmi, “Detecting cloud based phishing attacks using stacking ensemble machine learning technique,” *International Journal For Science Technology And Engineering*, vol. 11, no. 3, pp. 360–367, 2023.
- [12] C. Rajeswary and M. Thirumaran, “The lstm-based automated phishing detection driven model for detecting multiple attacks on tor hidden services,” *J. Intell. Fuzzy Syst.*, vol. 44, no. 6, p. 8889–8903, jan 2023.
- [13] B. Subba, “A heterogeneous stacking ensemble-based security framework for detecting phishing attacks,” in *2023 National Conference on Communications (NCC)*, 2023, pp. 1–6.
- [14] P. Švančárek, “Phishing attack detection using machine learning.” Lecture notes in networks and systems, 2023, pp. 301–312.

- [15] A. A. A. E.-S. M. E.-k. M. T. Diana T. Mosa, Mahmoud Y. Shams, “Machine learning techniques for detecting phishing url attacks,” *Computers, Materials & Continua*, vol. 75, no. 1, pp. 1271–1290, 2023.
- [16] S. Yu, C. An, T. Yu, Z. Zhao, T. Li, and J. Wang, “Phishing detection based on multi-feature neural network,” in *2022 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, 2022, pp. 73–79.
- [17] J. Novakovic and S. Marković, “Detection of url-based phishing attacks using neural networks,” *2022 International Conference on Theoretical and Applied Computer Science and Engineering (ICTASCE)*, pp. 132–136, 2022.
- [18] S. A. Salihu, I. D. Oladipo, A. A. Wojuade, M. Abdulraheem, A. O. Babatunde, A. R. Ajiboye, and G. B. Balogun, “Detection of phishing urls using heuristics-based approach,” in *2022 5th Information Technology for Education and Development (ITED)*, 2022, pp. 1–7.
- [19] J. Tanimu and S. Shiaeles, “Phishing detection using machine learning algorithm,” in *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*, 2022, pp. 317–322.
- [20] M. Sánchez-Paniagua, E. Fidalgo, E. Alegre, and R. Alaiz-Rodríguez, “Phishing websites detection using a novel multipurpose dataset and web technologies features,” *Expert Systems with Applications*, vol. 207, p. 118010, 2022.
- [21] R. D. Corin. (2019) Architecture of the wide and deep neural network. Accessed on 2024-05-09. [Online]. Available: https://www.researchgate.net/figure/Architecture-of-the-wide-and-deep-neural-network-The-prediction-of-the-sensors-states-is_fig1_337945667
- [22] Creative Commons Attribution 4.0 International. (2023) Tabnet architecture consisting of the encoder for classification. Accessed on 2024-05-09. [Online]. Available: <https://www.researchgate.net/figure/>

TabNet-architecture-consisting-of-the-encoder-for-classification-This-is-composed-of _
fig1_371301234

- [23] Creative Commons Attribution-NonCommercial 4.0 International. (2023) Deep neural network (dnn) model architecture. Accessed on 2024-05-09. [Online]. Available: https://www.researchgate.net/figure/Deep-Neural-Network-DNN-model-architecture_fig1_370641687
- [24] learnopencv. An example of fnn with one hidden layer. Accessed on 2024-05-09. [Online]. Available: <https://learnopencv.com/understanding-feedforward-neural-networks/>

Table A.1: Project Detail

Student Details

| | | | |
|---------------------|--------------------------|------------------|------------|
| Student Name | Ganesh S Nayak | | |
| Registration Number | 200911008 | Section/Roll No. | B/02 |
| Email Address | ganeshsnayak29@gmail.com | Phone No.(M) | 9591051320 |

Project Details

| | | | |
|----------------------|---|-------------------|------------|
| Project Title | Detecting Phishing Attacks using Machine Learning | | |
| Project Duration | 4 Months | Date of Reporting | 08-01-2024 |

Internal Guide Details

| | | | |
|---------------------------------------|--|--|--|
| Faculty Name | Dr. Balachandra | | |
| Full Contact Address with PIN Code | Department of Information and Communication Technology, Manipal Institute of Technology, Manipal-576104 | | |
| Email Address | bala.chandra@manipal.edu | | |