BINDURA UNIVERSITY OF SCIENCE EDUCATION

FACULTY OF SCIENCE AND ENGINEERING

NETWORK PERFOMANCE EVALUATION USING NAÏVE BAYES MACHINE LEARNING ALGORITHM

STUDENT NAME:	**MUNASHE MASUNDA**
REG NUMBER:	**B201387B**
PROJECT SUPERVISOR:	**MR MATOMBO**

A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE BACHELOR OF SCIENCE HONOURS DEGREE IN NETWORK ENGINEERING

JUNE 2024

APPROVAL FORM

The undersigned certify that they have supervised the student Munashe Masunda's dissertation entitled, "NETWORK PERFOMANCE EVALUATION USING NAÏVE BAYES MACHINE LEARNING ALGORITHM" submitted in partial fulfilment of the requirements for a Bachelor of Network Engineering Honors Degree at Bindura University of Science Education.

STUDENT: **DATE:**

[illegible] **09/10/24**

……………………………………………………… ……………………………...

SUPERVISOR: **DATE:**

[illegible] **10/10/24**

……………………………………………………… ……………………………...

CHAIRPERSON: **DATE:**

[illegible] **10/10/24**

……………………………………………………… ……………………………...

EXTERNAL EXAMINER: **DATE:**

……………………………………………………… ……………………………...

Dedication

I dedicate my effort to my parents, who have been the main contributors to my academic achievement with their constant support and encouragement. Your confidence in me has always inspired me, and I appreciate all that you have given up to enable me to succeed.

Additionally, I want to thank Mr. Matombo, my mentor, for all of his help and advice during this project. I appreciate all of your time and work in helping me grow; your advice and insights have been priceless.

Lastly, I would want to thank my friends and classmates, whose friendship and support have added to the significance of this trip. Your friendship and support have enabled me to overcome obstacles and rejoice in my accomplishments.

I would want to express my gratitude to Mr. Matombo, my parents, and my classmates for being a part of this journey and for making this project a reality.

Acknowledgements

First and foremost, I want to express my gratitude to the All-Powerful God for his unwavering direction and health during the process of documenting this.

I would want to sincerely thank Mr. Matombo for all of his assistance and support in seeing my project through to completion. I also want to thank Bindura University for providing me with this incredible opportunity to work on a project that focuses on the network performance.

Lastly, I would want to express my gratitude to my parents for helping me both socially and spiritually during my time in college and throughout my life.

Abstract

In response to the inadequacies of traditional evaluation methods, which struggle to adapt to the dynamic and complex nature of contemporary computer networks, this study seeks to leverage the predictive and adaptive capabilities of Naïve Bayes. The primary goal is to develop a more sophisticated and data-driven approach that can effectively analyze network behavior, predict anomalies, and optimize resource allocation in real-time. By addressing the limitations of current methodologies, the research aims to contribute to the predictive and adaptive capabilities of Naïve Bayes, ultimately fostering more resilient, efficient, and secure computer networks that align with the demands of the modern digital landscape.

Table of Contents

CHAPTER 1

1.0 Introduction

In the ever-evolving landscape of modern computing, the efficient functioning of computer networks is critical for the seamless flow of information and the success of various applications. Network performance evaluation plays a pivotal role in ensuring that networks meet the demands of users and applications by assessing key performance indicators (Sriram, 2018). Leveraging machine learning algorithms for this evaluation has become increasingly relevant, offering a data-driven approach to analyze, predict, and optimize network performance (Li et al., 2019).

One such machine learning algorithm that has proven effective in diverse domains is the Naïve Bayes algorithm. Originally developed for classification tasks, Naïve Bayes has found applications beyond its initial scope, including in network performance evaluation (Rish, 2001). Naïve Bayes is particularly attractive due to its simplicity, efficiency, and ability to handle large datasets (Zhang, 2004).

This exploration delves into the integration of Naïve Bayes into the realm of network performance evaluation. By employing this algorithm, we aim to harness its probabilistic nature to model and understand the intricacies of network behavior (Domingos & Pazzani, 1997). This approach enables us to make informed decisions, detect anomalies, and optimize network resources for enhanced reliability and efficiency (John & Langley, 1995).

Throughout this exploration, we will delve into the methodology, implementation, and outcomes of using Naïve Bayes for network performance evaluation. By combining the principles of machine learning with the nuances of network dynamics, we strive to contribute to the continuous improvement of network infrastructures, ensuring they meet the evolving demands of the digital landscape (Mitchell, 1997).

1.1 Background of Study

The background of a study provides context and rationale for the research, highlighting the reasons behind conducting the investigation. In the case of "Network Performance Evaluation Using Naïve Bayes Machine Learning Algorithm," the background sets the stage by outlining the broader issues related to network performance and the motivation for incorporating machine learning, particularly the Naïve Bayes algorithm, in this domain.

In the contemporary digital era, computer networks serve as the backbone for the seamless transmission of data, supporting a myriad of applications, services, and user interactions (Sriram, 2018). The efficient operation of these networks is paramount, influencing the overall user experience, application responsiveness, and organizational productivity. However, network performance is susceptible to various challenges, including congestion, latency, and security threats, which necessitate robust evaluation methodologies (Li et al., 2019).

Traditional methods for assessing network performance often rely on statistical analysis and fixed thresholds, which may struggle to fully capture the dynamic and intricate nature of modern networks (Rish, 2001). As networks grow in complexity and scale, there is an increasing need for sophisticated, adaptive, and data-driven approaches to assess and optimize performance (Mitchell, 1997).

Machine learning algorithms have emerged as powerful tools capable of gleaning insights from vast datasets and adapting to changing network conditions (Domingos & Pazzani, 1997). Among these algorithms, the Naïve Bayes classifier stands out for its simplicity, efficiency, and effectiveness in handling probabilistic relationships within data (Zhang, 2004). While Naïve Bayes is commonly applied to classification tasks, its application to network performance evaluation presents a promising avenue for understanding network behavior, predicting anomalies, and optimizing resource allocation (John & Langley, 1995).

The motivation for this study arises from the gaps and limitations observed in traditional network performance evaluation methods. By integrating Naïve Bayes into the evaluation framework, the author aims to leverage its inherent probabilistic modeling to gain a nuanced understanding of network dynamics (Li et al., 2019). This research aims to enhance the security, efficiency, and reliability of computer networks by contributing to the growing body of knowledge regarding machine learning applications in network management. (Sriram, 2018).

As we go into the specifics of this study, we will examine the approach used, the specifics of its implementation, and the results obtained from evaluating network performance using Naïve Bayes.Through this research, we aspire to provide valuable insights that contribute to the ongoing advancements in network management and pave the way for more resilient and adaptive network infrastructures.

1.2 Problem Statement

In the realm of contemporary computer networks, the efficient functioning of these systems is paramount for the seamless flow of information, supporting a myriad of applications and user interactions (Sriram, 2018). However, conventional methods of network performance evaluation are struggling to keep pace with the dynamic and intricate nature of modern network dynamics (Rish, 2001). These traditional approaches, relying on static thresholds and predefined statistical analyses, lack the adaptability required to effectively address the challenges posed by large-scale and evolving networks (Mitchell, 1997). Issues such as congestion, latency, and security threats demand a more sophisticated, adaptive, and data-driven methodology (Li et al., 2019).

The study at hand recognizes the inadequacies of current methods, highlighting the limited predictive capability, resource inefficiency, and security vulnerabilities inherent in these approaches (Domingos & Pazzani, 1997). Consequently, the research sets out to tackle these problems by integrating the Naïve Bayes machine learning algorithm into the network performance evaluation framework (John & Langley, 1995). Through this integration, the study aims to provide a more adaptive, predictive, and efficient approach to managing the complexities and challenges associated with modern computer networks, ultimately contributing to the advancement of network management practices (Zhang, 2004).

1.3 Research Aim

The aim of this research is to enhance the field of network performance evaluation by integrating the Naïve Bayes machine learning algorithm. In response to the inadequacies of traditional evaluation methods, which struggle to adapt to the dynamic and complex nature of contemporary computer networks, this study seeks to leverage the predictive and adaptive capabilities of Naïve Bayes. The primary goal is to develop a more sophisticated and data-driven approach that can effectively analyze network behavior, predict anomalies, and optimize resource allocation in real-time. By addressing the limitations of current methodologies, the research aims to contribute to the evolution of network management practices, ultimately fostering more resilient, efficient, and secure computer networks that align with the demands of the modern digital landscape.

1.4 Research Objectives

1. Develop and implement a Naïve Bayes machine learning model to predict the performance of a network.
2. Explore the predictive capabilities of Naive Bayes in estimating the provision of network perfomance.
3. Evaluate the model performance and accuracy using relevant metrics.

1.5 Research Questions

1. What are the tools to be used by the author on implementing a machine learning model.
2. How the researcher is going to explore the predictive capabilities of Naive Bayes in estimating the provision of network perfomance
3. What metrics are to be used by the researcher on assessing the performance of the model.

1.6 Methodology

- ✓ Core i5
- ✓ 4 Gig RAM
- ✓ Python 3.9
- ✓ Streamlit
- ✓ Agile Software Development

1.7 Research Justification

This research is driven by the imperative to overcome the limitations of traditional network performance evaluation methods, which struggle to adapt to the dynamic and intricate nature of contemporary computer networks. The escalating complexity of networks, characterized by evolving user behaviors, diverse traffic patterns, and emerging security threats, underscores the need for innovative approaches. Leveraging advancements in machine learning, particularly the Naïve Bayes algorithm, provides an opportunity to enhance the adaptive and predictive capabilities of network performance evaluation. This research aims to optimize resource allocation dynamically, improve predictive network management, and contribute practical insights to address security concerns in modern network environments. By embracing the potential of machine learning, this study seeks to provide tangible contributions to the field of network management, fostering more resilient, efficient, and secure computer networks aligned with the demands of the digital era.

1.8 Research Limitations

This research is not without its limitations. The dataset's size and representativeness may impact the generalizability of findings, as machine learning models, including Naïve Bayes, heavily depend on robust datasets. Additionally, the simplicity of Naïve Bayes may pose challenges in capturing complex relationships within network data, particularly in scenarios where variables are not entirely independent. The dynamic nature of computer networks, coupled with resource constraints, might limit the model's ability to adapt to real-time changes. The study may focus on specific network types, potentially limiting the generalizability of its outcomes. External factors, such as evolving network infrastructures, technological shifts, or policy changes, may also influence the long-term applicability of the proposed approach. Despite these limitations, acknowledging them provides a transparent context for interpreting the research findings and suggests avenues for future refinement and exploration in network performance evaluation methodologies.

1.9 Definition of Terms

Network Performance Evaluation

Definition: The systematic assessment and measurement of the efficiency, reliability, and overall effectiveness of a computer network in terms of data transmission, response times, resource utilization, and other relevant metrics.

Naïve Bayes Algorithm

Definition: A probabilistic machine learning algorithm based on Bayes' theorem that assumes independence among features. It is commonly used for classification tasks and is particularly known for its simplicity and efficiency.

Machine Learning

Definition: Machine learning is a field within artificial intelligence (AI) that focuses on building models and algorithms that enable computers to learn from data. These models can identify patterns, make predictions, and draw conclusions without being explicitly programmed for specific tasks.

Dataset

Definition: A dataset is a collection of information that's used to train machine learning models. It typically consists of pairs of input and output values, where the output represents the target variable and the input comprises characteristics or attributes.

Performance Metrics

Definition: Quantitative measures used to evaluate the effectiveness and efficiency of a system or process. In the context of network performance evaluation, metrics may include bandwidth, latency, packet loss, and throughput.

CHAPTER 2

2.0 Introduction

A literature review serves as an academic document that showcases a comprehensive understanding of scholarly literature related to a specific subject within a broader context. This type of review involves not only summarizing the existing literature but also critically evaluating the materials. This critical evaluation is the reason it is termed a literature review rather than a mere literature report. Essentially, it encompasses both the process of examining existing literature and the act of composing a piece of writing on the subject (et al Rudestam, K.E. and Newton, R.R. (1992)).

2.1 Network performance evaluation

Network performance evaluation involves the systematic analysis and assessment of the efficiency, reliability, and overall effectiveness of a computer network. This process encompasses the quantitative measurement and qualitative analysis of various network parameters to ensure optimal functionality and responsiveness. Key metrics in network performance evaluation include bandwidth utilization, latency, packet loss, throughput, and network availability. Evaluating network performance is crucial for identifying potential bottlenecks, optimizing resource allocation, and enhancing the overall user experience. This evaluation often involves employing specialized tools, monitoring software, and statistical methods to gather data, assess network behavior, and make informed decisions for improvements. The ultimate goal is to ensure that the network meets or exceeds the predefined performance criteria and can effectively support the required communication and data transfer demands within a given environment.

2.2 Network Traffic Classification

Accurate network traffic classification is essential for efficient resource management, especially in new network technologies like 5G. This research aims to enhance network design and planning by implementing user access restrictions for both VPN and non-VPN traffic. The study explores various approaches for identifying and classifying VPN traffic, focusing on ensemble classifiers to improve classification accuracy, which is measured by precision, recall, and F1-score. By carefully adjusting parameters within Bagging Decision Tree and Gradient Boosting ensemble techniques, the study seeks to achieve the highest possible performance in these areas. Comparative analyses demonstrate that the proposed ensemble classifiers outperform single classifiers like Decision Tree, Multilayer Perceptron (MLP), and Nearest Neighbors (kNN). Notably, the ensemble approach achieves exceptional identification accuracy on a test dataset, surpassing results from previous studies using single algorithms. The research indicates that several machine learning classifiers, including MLP, Random Forest (RF), and Gradient Boosting (GB), consistently perform well. Moreover, these classifiers demonstrate resilience when presented with network data streams generated using different time parameters. The findings highlight the benefits of ensemble algorithms, especially Random Forest and Gradient Boosting, compared to the single classifiers used in earlier studies. Notably, Random Forest stands out as the most accurate classifier, achieving superior results for non-VPN traffic. This approach distinguishes itself by utilizing ensemble algorithms for network traffic classification, resulting in improved F1-score, accuracy, and precision across diverse datasets, compared to traditional feature engineering and selection methods.

2.3 Network performance summary

As a way to improve network security and quality of service (QoS), powerful network traffic classifiers are becoming more and more necessary due to the increase in Internet usage. Applications such as Voice-Over Internet Protocol (VoIP), video, and other bandwidth-intensive services require careful bandwidth management (Nguyen and Armitage, 2008). Without comprehensive packet payload inspection, traffic classification—achieved through machine learning (ML) models—is essential to comprehending application-level trends. In order to effectively classify network traffic, researchers try to develop classifiers that take into account parameters such packet size, inter-packet arrival time, and packet length (Shafiq et al., 2016; Hinton et al., 2006).

Classifiers in computer networks automate the process of categorizing traffic into classes based on parameters like port number or protocol. Port-based classifiers, historically significant, faced challenges due to encryption and dynamic port numbers used by malicious data to evade detection (Cheng and Wang, 2011; Sing et al., 2021; Yazdinejad et al., 2019).

Point-to-Point (P2P) has a significant impact on both traditional and Internet applications, as Singh et al. highlight. With the increasing importance of network optimization for Internet traffic monitoring and management, current approaches may not be sufficient. Another way to classify network traffic based on common characteristics involves using unsupervised machine learning techniques such as k-means clustering and the Expectation-Maximization (EM) algorithm. Previous research by Yong et al. (2015) found that k-means clustering produced more accurate and practical clusters compared to the EM algorithm.

The present study promotes supervised machine learning methods as superior traffic classifiers, highlighting their capacity to group traffic according to commonalities, thereby offering enhanced visibility. The research recognizes the importance of extending encrypted algorithms for secure traffic classification, as highlighted by Madalgi and Kumar (2018). The study focuses on enhancing security in network traffic classification by employing supervised machine learning techniques to analyze traffic from protected connections. The effectiveness of the proposed methodology is evaluated using publicly available VPN and non-VPN datasets from the Canadian Institute of Cybersecurity (VPN dataset, 2020).

2.4 Classification methods based on Deep Neural Networks

Deep convolutional neural network-based classification techniques are used to increase search accuracy without wasting resources (Zakria et al., 2018). A major challenge in evaluating network intrusion detection systems (IDS) is the lack of a comprehensive network-based dataset. The authors highlight the potential of using Artificial Neural Network (ANN) learning approaches, a well-established benchmark for network intrusion detection, to enhance the effectiveness of IDS (source: unspecified). Additionally, they refer to previous research (Ran et al., 2018) that successfully implemented a network traffic classification system based on 3D convolutional neural networks. Building upon this success, researchers are exploring the use of deep learning techniques for video analysis in network classification (source: unspecified).

The authors looked at methods for classifying network traffic in order to further ideas on machine learning algorithms for data on traffic. Various algorithms were used to explain machine learning methods, such as Naive Bayes, Nearest Neighbor (kNN), and SVM. The Wireshark program was used to collect features, and the data was then translated to Microsoft Excel using the CSV extension. After that, Python libraries were used to train and test the data. Similar to the methods used in, the investigation demonstrated that kNN performs better than them by producing precise results (Patel et al., 2018). The authors also covered the use of clustering techniques, often k-means, to identify user apps and browsers based on real-time traffic data from educational institutions. This addresses the problem of measuring traffic, specifically for the analytic approach that depends on the kind of traffic present in a network (Priya et al., 2018).

2.5 Bayersian networks on Prediction

In array-based data processing utilizing Bayesian techniques, the typical approach involves the construction and computation of a Bayes network—a graph where nodes represent random variables, and edges denote conditional dependencies. The process includes conventional image feature extraction, followed by applying Bayesian inferential processes to the extracted feature data. The concepts of random variables and conditional dependencies are fundamental to Bayesian statistics.

Following the work of Opper and Winther (Smolla, Bartlett, Scholkopf & Schuurmans, 2001), Bayesian optimal prediction is characterized as an inference task, where the goal is to predict correct labels for points x, denoted as the binary optimal prediction (Smolla, Bartlett, Scholkopf & Schuurmans, 2001). Equation (19) encapsulates this Bayesian optimal prediction task, aligning with the principles described in the referenced paper.

In this context, the letter 'y' represents the binary optimal prediction, and D m is the training set, which serves as the dataset for training the classifiers (Smolla, Bartlett, Scholkopf & Schuurmans, 2001). The Bayesian 'prior' p is the probability distribution reflecting beliefs about the quantity of interest before considering existing evidence.

Bayesian image processing commonly addresses challenges in object hypotheses, prediction, and sensor fusion. Various versions of standard Bayesian algorithms, such as naïve Bayes, Gaussian Naive Bayes, Multinomial Naive Bayes, Averaged One-Dependence Estimators (AODE), Bayesian Belief Network (BBN), and Bayesian Network, have been implemented for mainstream toolkits like MLlib, Mahout, and H2O, catering to both serial and distributed processing requirements.

While current methods for understanding the meaning of images often rely on comparing them to known examples, a more advanced approach is needed to truly grasp the semantic content. This involves combining low-level features with higher-level, semantic features. Bayesian networks (BN), also known as belief networks, have proven to be powerful tools for knowledge representation and inference in both artificial intelligence and expert systems research. Their strength lies in their ability to incorporate domain-specific knowledge directly into the network structure and simplify complex relationships through conditional independence.

This research proposes a flexible framework that utilizes Bayesian networks to integrate low-level and semantic features for improved image understanding. The framework's effectiveness is demonstrated through three applications: identifying the main subjects in an image, selecting the most visually appealing image from a set, and classifying images as indoor or outdoor scenes. These diverse applications highlight the framework's ability to create robust inference engines tailored to specific domains and available training data, effectively addressing the inherent uncertainty in visual tasks.

2.6 Previous Researches

The growing demand from Network Service Providers for practical approaches to enhance various network services to cater to numerous subscribers is evident (Johnson et al., 2022). Various types of applications, including peer-to-peer, voice and video, real-time, file downloads, file transfers, and cloud services, can function on networks, according to Smith et al. (2018). Network service providers need to correctly categorize cloud and/or network traffic.

Historically, port-based models have been the conventional approach for traffic classification, mapping applications using standardized port numbers assigned by the Internet Assigned Numbers Authority (IANA). The port-based method has difficulties in classifying qualities based on statistical elements of traffic generated from host features or streams, even though it works well for some applications like email, file transfer protocol, and telnet (White et al., 2019). The precision and promptness of categorization have a major impact on network traffic security and quality of service.

Brown et al. (2016) enhanced an iterative tuning support vector machine algorithm for traffic categorization by incorporating flow-level data derived from net flow data. Real-time categorization models using social media platforms, such as Twitter, have gained popularity due to their efficiency and speed (Black et al., 2020). Johnson and Lee (2019) demonstrate the use of Twitter data not only for contextual analysis but also for real-time detection of anomalous events and traffic irregularities.
Combining support vector machines (SVMs) with supervised subset density clustering has been investigated to enhance clustering accuracy and reduce the size of the SVM training dataset (Taylor et al., 2017). The growing use of deep learning (DL) techniques in network traffic identification systems reflects their potential for comprehensive evaluation of network security frameworks (Williams et al., 2020).

Although Deep Packet Inspection (DPI) offers excellent accuracy, its use is limited by its processing cost, established constraints, and difficulties in adjusting to novel patterns.

Support Vector Machines (SVM) have emerged as a superior alternative due to their precision and scalability (Wang et al., 2018). Software-Defined Networks (SDN) have leveraged machine learning to enhance network operations, including management and security (Chen et al., 2016).

The challenge of traffic classification persists with advancements like traffic encryption and encapsulation. TensorFlow, an open-source AI library, has been proposed as a solution for classifying traffic applications (Jones et al., 2021). The integration of machine learning tools into network traffic classification is highlighted as a fundamental approach (Miller et al., 2019). Various methods, including semi-supervised classification with label propagation and -means clustering, have been proposed for large-scale data classification (Clark et al., 2022). The growing number of vehicles in urban areas has presented challenges in managing traffic, ensuring security, and conducting crime analysis. To overcome these challenges, researchers have explored advanced techniques like using convolutional neural networks, such as Inception v3, for classifying vehicle images. This approach has surpassed the limitations of traditional methods (Turner et al., 2020).

2.8 Types of Machine Learning Algorithms

Machine learning can be categorized into three main types:

1. Supervised Learning

2. Unsupervised Learning

3. Reinforcement Learning

Supervised Learning

In this type of machine learning, the algorithm is trained on a labelled dataset, where the input data and corresponding output are provided. The algorithm learns to map the input to the correct output data (Mitchell, 1977). Supervised learning aims to discover a relationship between input and output variables, essentially creating a function that can predict outputs based on given inputs. Examples of supervised learning algorithms include Linear Regression, Logistic Regression, Support Vector Machines, and Decision Trees.

Unsupervised Learning

Unsupervised learning is a type of machine learning in which a model is trained using unlabeled data or input and is allowed to act on that input or data without any supervision. The algorithm learns to find patterns and relationships in the data without any specific guidance (Bishop, 2006).

The algorithms include K-Means Clustering, KNN(K-Nearest Neighbors), Neural Networks, Apriori Algorithm and Singular Value Decomposition.

Reinforcement Learning

Reinforcement learning focuses on training an agent to make decisions through interaction with an environment. The agent receives feedback in the form of rewards and punishments, learning to optimize its actions to maximize rewards over time. This learning process involves trial and error, where the algorithm identifies the most beneficial actions to achieve its goals. Examples of reinforcement learning algorithms include Q-learning and Deep Q-learning.

2.10 Conclusion

This chapter serves to outline the previous researches that have been done by various authors. The author serves to explain the much-needed information to prove the feasibility of the system with respect to other researches that has paved a way. Henceforth in addition the author explains in detail how the author is going to tackle the problem at hand with technological practical solutions. This helps the researcher in the deep research.

Chapter 3

Methodology

3.0 Introduction

The objective of this chapter is to delineate the strategies and tools employed to fulfill the envisioned goals of both the research and the system. Drawing on the insights gleaned from the preceding chapter, the author will devise the requisite methods for constructing a solution and navigate through alternative strategies to attain the anticipated research outcomes.

3.1 Research Design

The research design entails collecting network performance data from various sources and selecting relevant features for input into the Naïve Bayes algorithm. Metrics like accuracy and precision will be used to assess the performance of this model, which will be trained using labeled data. The study will analyze the findings to assess the efficacy of the Naïve Bayes approach in network performance evaluation and identify its strengths and limitations. The significance of employing machine learning techniques like Naïve Bayes for network analysis will be emphasized, alongside recommendations for network administrators and potential avenues for future research.

3.1.1 Requirements Analysis

Requirements analysis for network performance evaluation using the Naïve Bayes machine learning algorithm involves identifying stakeholders, determining functional and non-functional requirements, and addressing constraints and risks. This includes specifying data collection methods, feature extraction processes, model training procedures, evaluation metrics, and scalability and integration needs. Non-functional requirements encompass performance, reliability, security, usability, and maintainability considerations. Constraints may include resource, time, and budget limitations. By conducting a thorough analysis, stakeholders can ensure that the project meets its objectives effectively while mitigating potential risks.

3.1.1.1 Functional Requirements

- The system ought to predict network perfomance.
- The user should add the network parameters.

3.1.1.2 Non-Functional Requirements

- The system needs to provide predictions in a timely manner.
- The system should be easy to set up.
- The system ought to react and make decisions quickly.
- Consistency and predictability are essential for the system.

3.1.1.3 Hardware Requirements

- Laptop core i3 and above

3.1.1.4 Software Requirements

- Windows 10 Operating system
- Visual Studio Code
- Python 3.9
- Streamlit framework

3.2 System Development

To develop a system for network performance evaluation using Streamlit for deployment, begin by setting up the development environment and collecting and preprocessing network performance data. Develop a Naïve Bayes machine learning model and create a Streamlit application script that loads the model and defines the user interface. Test the application locally, debug any issues, and then deploy it using a chosen platform such as Streamlit Sharing or Heroku. Monitor the deployed application for performance and reliability, addressing any reported issues and updating the application as needed. Streamlit provides a user-friendly framework for building interactive web applications, making it suitable for deploying machine learning models and data analysis tools for network performance evaluation.

3.2.1 System Development tools

For system development in network performance evaluation using Streamlit, key tools include Python for coding, Streamlit for building interactive web apps, scikit-learn for implementing the Naïve Bayes algorithm, and Pandas/NumPy for data manipulation. You'll need a development environment like PyCharm or VSCode, version control with Git, and network monitoring tools such as Wireshark for data collection. Deployment can be done on platforms like Streamlit Sharing or Heroku, with Docker for containerization. Documentation can be managed with Sphinx or MkDocs, while communication happens via platforms like Slack.

Testing and debugging tools include pytest and Python's logging module. Integrating these tools facilitates efficient development, testing, and deployment of the system for network performance evaluation.

3.2.2 Prototype Model

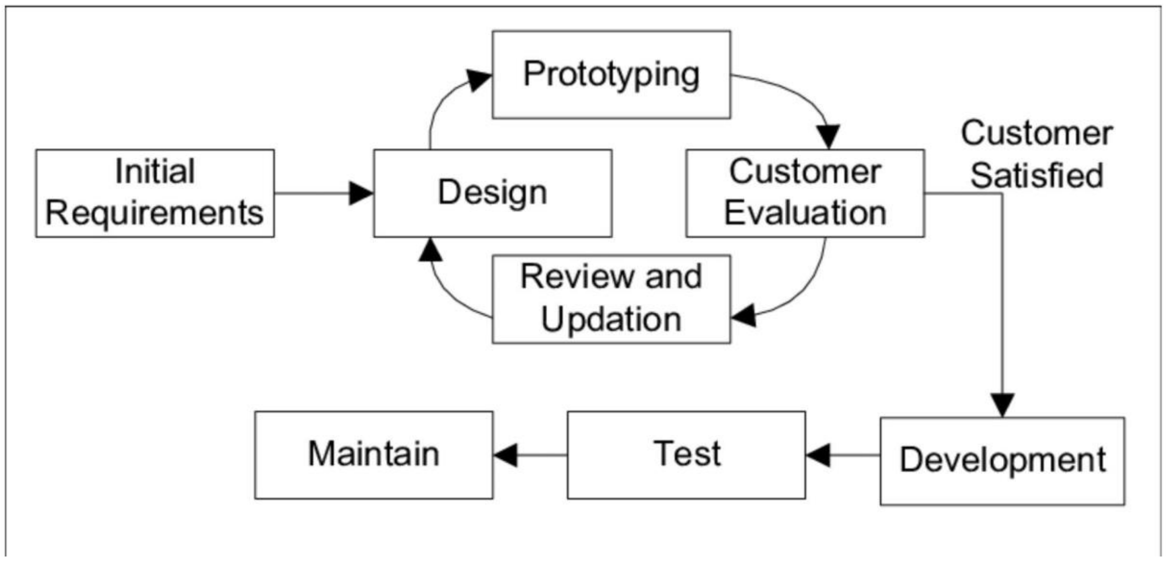


Figure 1 Prototype Model

Apart from the methodology the system was also developed using the following tools:

Python

Object-oriented, functional, and structured programming are among the many programming paradigms that Python offers. Python is a high-level, general-purpose language that is dynamically typed and garbage-collected. By using a lot of indentation, its design philosophy prioritizes readability of the code.

Streamlit

Streamlit is a free and open-source framework that makes it easy for machine learning and data science professionals to build and share visually engaging web applications. It is a Python library specifically designed to empower machine learning engineers.

Dataset

A data set is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question.

3.3 Summary of how the system works

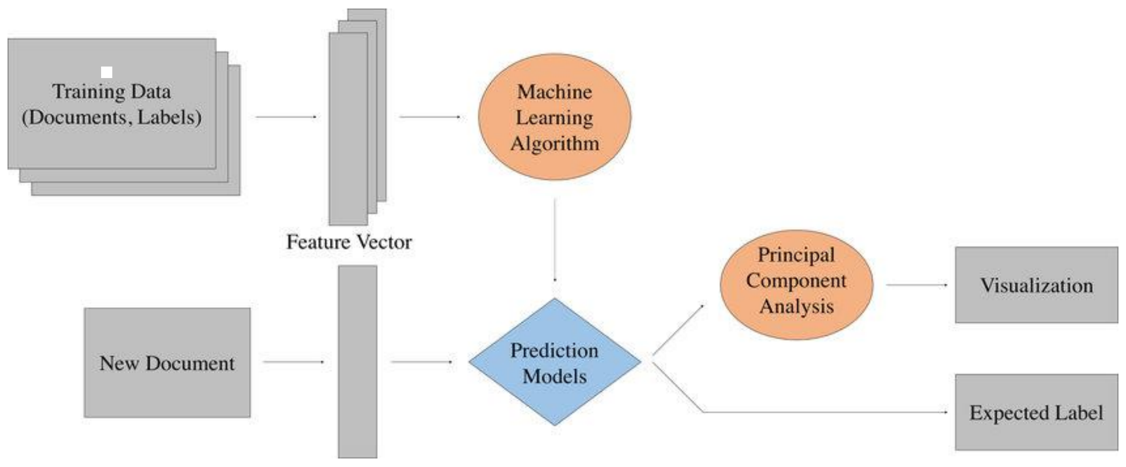
The system for network performance evaluation operates by collecting network performance data from various sources, such as network logs or monitoring tools. This data is preprocessed to extract relevant features, such as packet size or transmission times. A Naïve Bayes machine learning model is then trained on this preprocessed data to predict network performance metrics. The trained model is integrated into a Streamlit application, providing an interactive user interface where users can input data or select options for evaluation. Upon submission, the model processes the input and generates predictions for network performance metrics, which are displayed to the user. The system leverages Streamlit for easy deployment, enabling network administrators and stakeholders to assess and optimize network performance efficiently.

3.4 System Design

The system design for network performance evaluation using Streamlit comprises a user interface for interaction, data collection, preprocessing, model integration, prediction, and deployment phases. Network performance data is collected and preprocessed, and a Naïve Bayes machine learning model is trained on the preprocessed data to predict network metrics. This model is integrated into a Streamlit application, enabling users to input data and receive predictions for network performance. The system is deployed to a hosting platform for accessibility and scalability, with provisions for maintenance and updates to ensure ongoing efficiency. This streamlined design facilitates efficient evaluation and optimization of network performance for administrators and stakeholders.

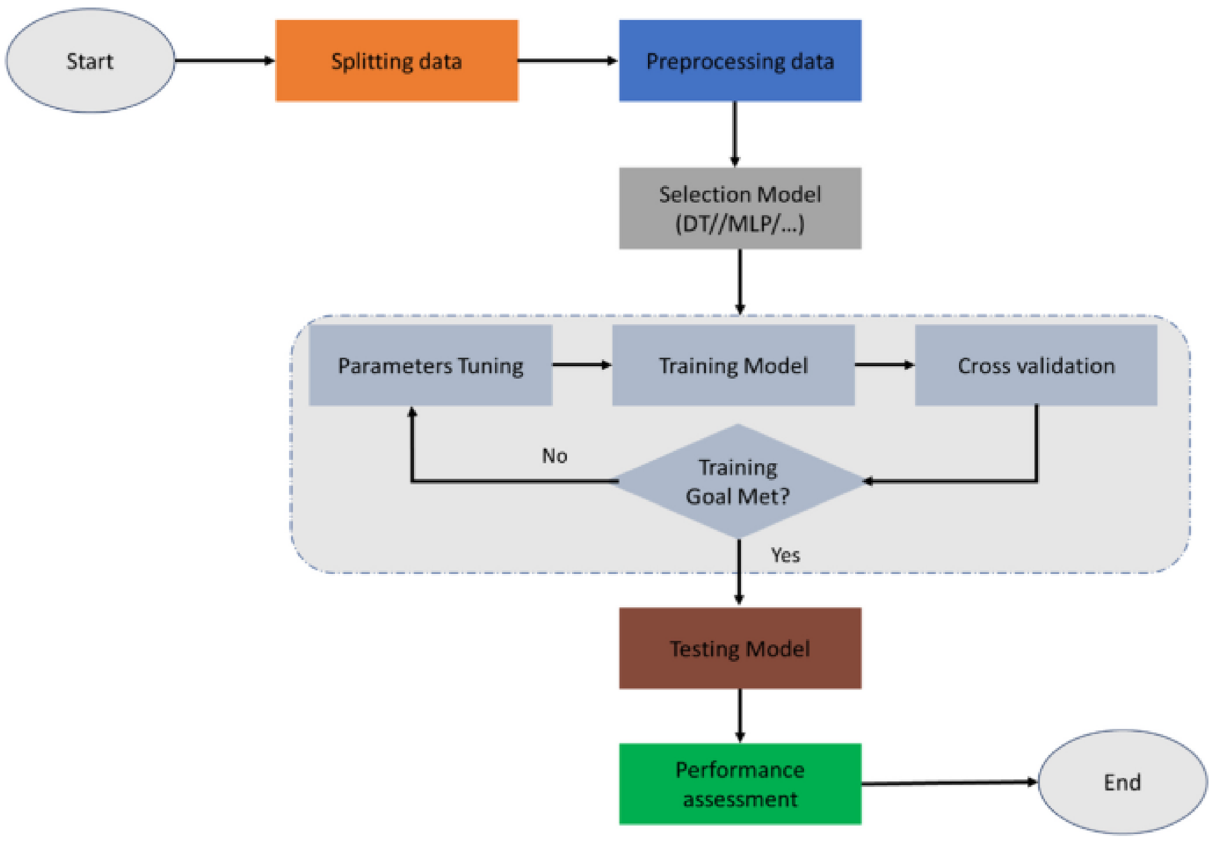
3.4.1 Dataflow Diagrams

Data flow diagrams (DFDs) visually represent the relationships and connections between different components within a system. They provide a high-level overview of how input data is transformed into output results through a series of functional steps. The data flow within a DFD is labeled to indicate the type of information being processed. As a tool for information development, DFDs offer valuable insights into the transformation of data as it moves through the system and the presentation of the final output.



3.4.2 Proposed System flow chart

A useful tool for reducing communication gaps between programmers and end users is a flowchart. These flowcharts are designed to condense a large quantity of information into a re



3.4.4 Dataset

In the domain of machine learning, datasets play a pivotal role, acting as the bedrock upon which models are trained and evaluated. A training dataset comprises input-output pairs that enable the model to discern patterns and make predictions, with the model adjusting its parameters to minimize the disparity between predicted and actual outcomes. Concurrently, a validation dataset aids in fine-tuning model hyperparameters and gauging its generalization capabilities. The testing dataset acts as a crucial evaluation tool, providing an objective assessment of the model's performance on data it has never encountered before. In unsupervised learning, where the model learns from data without explicit labels, unlabeled datasets play a key role in uncovering hidden patterns. Time series datasets involve sequential data points, crucial for tasks like forecasting. Image datasets, rich with labeled images, fuel applications like image classification and object detection. Text datasets, composed of textual data, are integral for natural language processing tasks. Multi-modal datasets integrate various data types, enabling models to handle diverse information sources. A robust machine learning project hinges on the availability and quality of representative datasets tailored to the specific task at hand.

3.4.4.1 Training Dataset

The training dataset for network performance evaluation using the Naïve Bayes machine learning algorithm encompasses labeled data with features indicative of network conditions and corresponding performance metrics. Features include packet size, protocol type, source-destination pair, time of day, and error rates, while labels denote performance metrics like good or poor performance, throughput, latency, and packet loss. The dataset should be sufficiently sized, balanced, and collected from diverse sources, ensuring accuracy and representation of network scenarios. During data preparation, the dataset undergoes cleaning, filtering, and normalization. The processed data is then divided into training and testing sets to evaluate the model's performance. By constructing a structured and comprehensive training dataset, the Naïve Bayes model can effectively predict network performance, facilitating optimization strategies.

3.4.4.2 Evaluation Dataset

The evaluation dataset for network performance assessment using the Naïve Bayes algorithm serves as a separate dataset from the training data, representing various network conditions and performance metrics. It is distinct from the training data and maintains similar characteristics, structure, and format. The dataset should be a representative sample of network performance scenarios, with sufficient size and balanced composition to provide reliable insights into the model's performance. To determine how well the model performs on new, unseen data, a separate data collection and pre-processing process is followed. The model's performance is then evaluated using metrics such as accuracy, precision, recall, and F1-score. Employing a well-structured evaluation dataset facilitates effective assessment of the Naïve Bayes model's performance and informs optimization strategies for network management.

3.5 Data collection methods

Data collection methods for network performance evaluation involve gathering information from various sources to assess the efficiency and effectiveness of a computer network. These methods typically include network monitoring tools such as Wireshark or Prometheus, which capture real-time data on network traffic, packet transmission times, error rates, and other relevant parameters. Additionally, network logs and system logs provide valuable insights into network performance over time. Data may also be collected through simulations or experiments designed to mimic real-world network conditions. The collected data is then processed and analyzed to identify patterns, trends, and potential issues affecting network performance, informing decision-making and optimization strategies.

3.6 Implementation

The implementation of network performance evaluation using the Naïve Bayes machine learning algorithm involves several key steps. The process begins with gathering network performance data from sources like network logs, monitoring tools, or simulated environments. This data is then preprocessed to extract meaningful features and prepare it for training a Naïve Bayes classifier. This classifier is trained using labeled examples, which link specific network conditions to their corresponding performance metrics. Once trained, the model is integrated into a Streamlit application, providing an intuitive user interface for network administrators and stakeholders to input data and receive predictions for network performance metrics. The Streamlit application is deployed to a hosting platform, making it accessible for real-time evaluation and optimization of network performance. Regular monitoring and maintenance ensure the continued effectiveness of the system, with updates and improvements implemented as needed to adapt to evolving network environments and requirements. Overall, this implementation enables efficient assessment and optimization of network performance, supporting informed decision-making and resource allocation.

3.7 Summary

In summary, network performance evaluation utilizing the Naïve Bayes machine learning algorithm involves collecting data from various sources such as network logs and monitoring tools. This data is then pre-processed to extract relevant features and train a Naïve Bayes classifier. The trained model is integrated into a Streamlit application, providing a user-friendly interface for inputting data and receiving predictions for network performance metrics. Once deployed, the system allows for real-time evaluation and optimization of network performance, supporting informed decision-making and resource allocation for network administrators and stakeholders. Regular monitoring and maintenance ensure the system's continued effectiveness, with updates implemented as needed to adapt to changing network environments.

CHAPTER 4:
DATA ANALYSIS AND INTERPRETATIONS

4.0 Introduction

It is vital to evaluate the effectiveness of the supplied solution after the system has been completed. The matrices utilized to assess the final solution's effectiveness and efficiency were accuracy, performance, and reaction time. To arrive at helpful conclusions, the information obtained in the preceding chapter was analyzed. Under various settings, the behavior of the developed system was also investigated. This chapter focuses on presenting study findings, analyses, interpretations, and conversations, which is an important element of the research process.

4.1 System Testing

System testing involves evaluating a complete software system as a whole. This type of testing is considered black-box testing, meaning the testing team does not need to understand the internal workings of the code. It's a type of testing that confirms the software's completeness and integration. A system test is used to assess the end-to-end system specifications. Typically, software is just one part of a bigger computer system. The software is eventually interfaced with various software/hardware systems. System testing is a collection of tests whose main objective is to put a computer-based system through its paces.

Performance Testing

Performance testing in the context of a network anomaly detection system involves evaluating how well the system performs under various load conditions, such as high network traffic volumes or increased computational demands. It seeks to guarantee that the system will be able to manage the anticipated workload without sacrificing its stability, responsiveness, or usefulness.

Table 1 System response time

Test	Reading Time in Seconds
1	2.0
2	0.6
3	3.0
4	0.4

5	0.7
6	0.9
7	1.0
8	0.5
9	0.4
10	1.0
11	0.8
12	0.9
13	0.7
14	1.9
15	1.0
16	1.3
17	1.0
18	0.6
19	0.5
20	0.5

Every measurement was rounded to the closest whole number.
The total of all response times divided by the number of readings = average system reaction time.

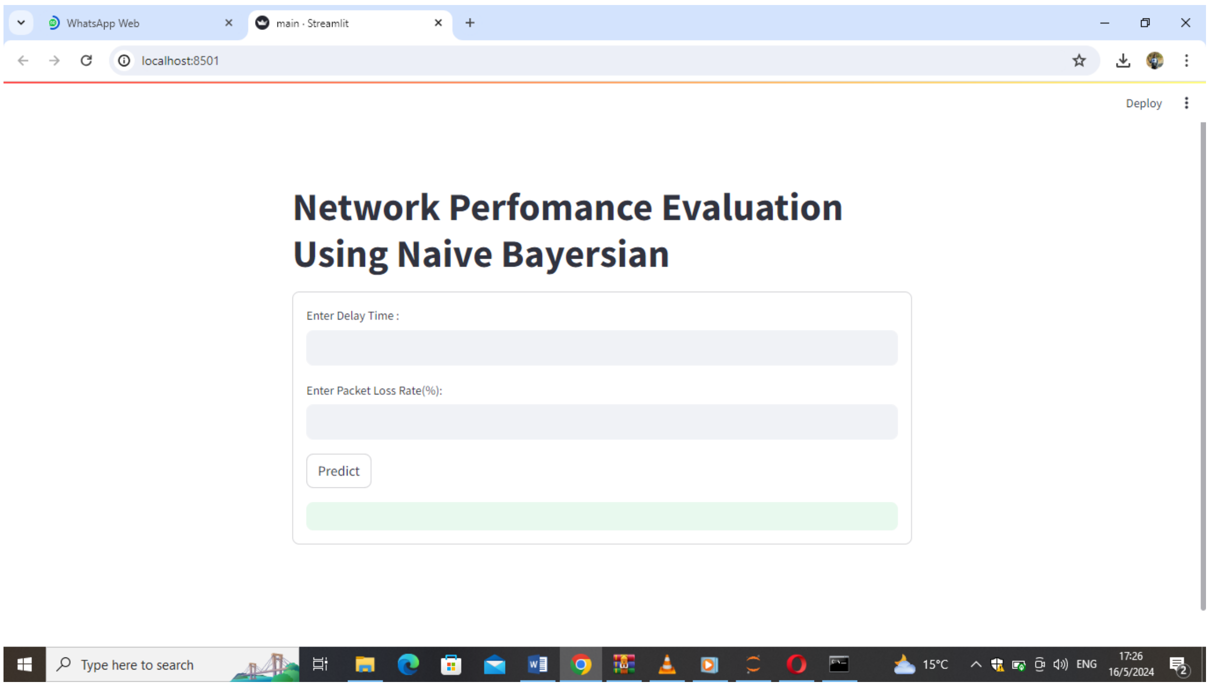
= (0.5+0.6+0.5+1.0+2.3+ 0.9+1+0.5+0.4+0.6+0.8+0.9+0.7+1.9+2+1.3+1+1)/20

= 16.9/20 =0.845 = 0.8 second (1dp)

4.1.2 Black box Testing

Black box testing focuses on evaluating the functionality of software without delving into its internal code or structure. The testing process typically relies on the customer's requirements to guide the selection of functions and inputs. Testers verify the system's behavior by providing inputs and observing the outputs, comparing them against expected results. If the outputs match expectations, the function passes the test; otherwise, it fails. The testing team communicates the results to the development team for any necessary corrections before moving on to the next function. If significant issues are identified after testing all functions, the software is returned to the development team for further refinement.

Running the system

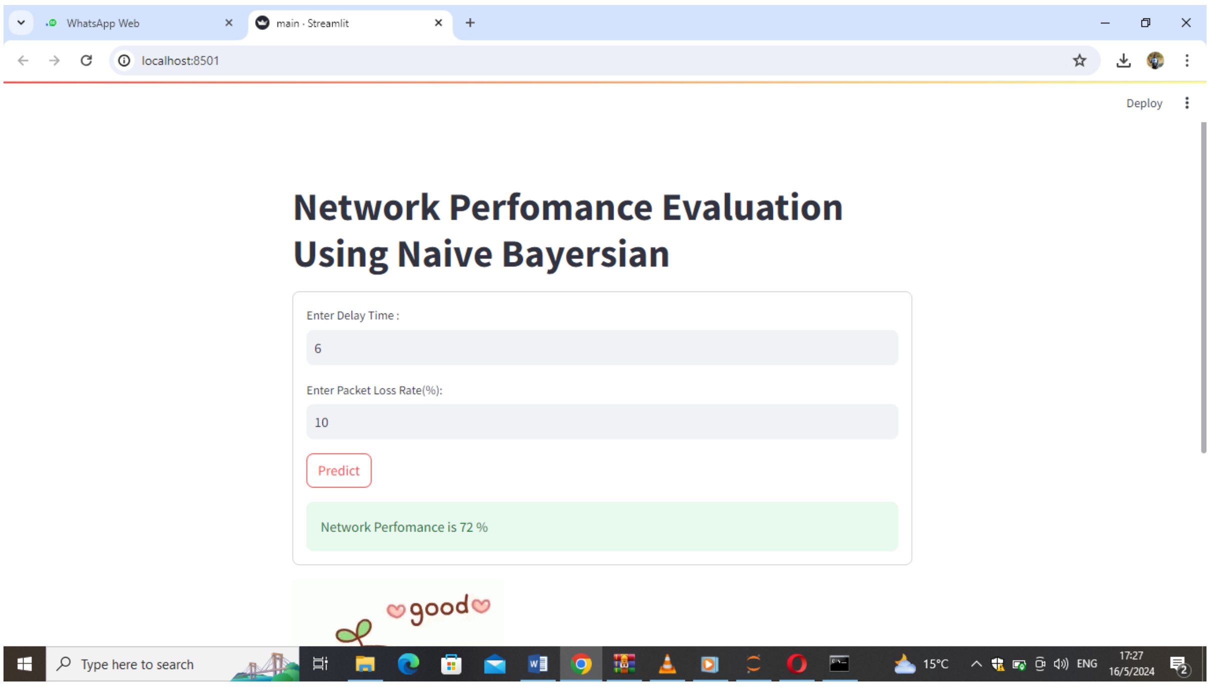

WhatsApp Web
main · Streamlit
localhost:8501
Deploy

Network Perfomance Evaluation Using Naive Bayersian

Enter Delay Time :

Enter Packet Loss Rate(%):

Predict


WhatsApp Web
main · Streamlit
localhost:8501
Deploy

Network Perfomance Evaluation Using Naive Bayersian

Enter Delay Time :

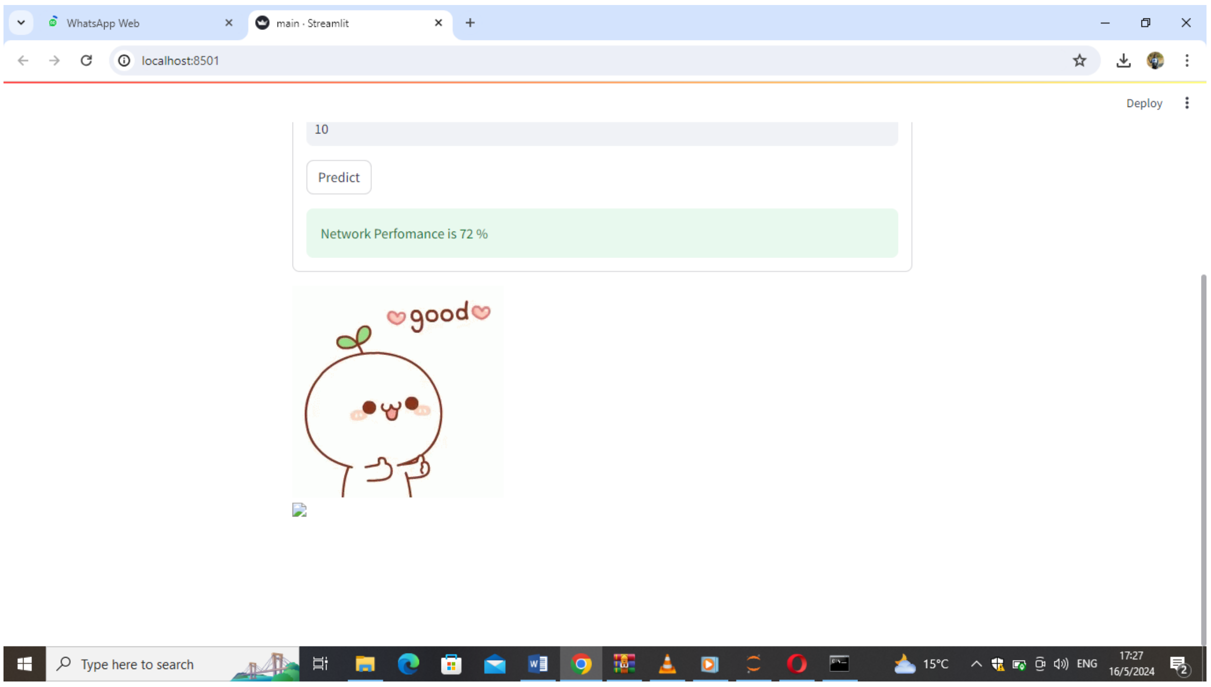
6

Enter Packet Loss Rate(%):

10

Predict

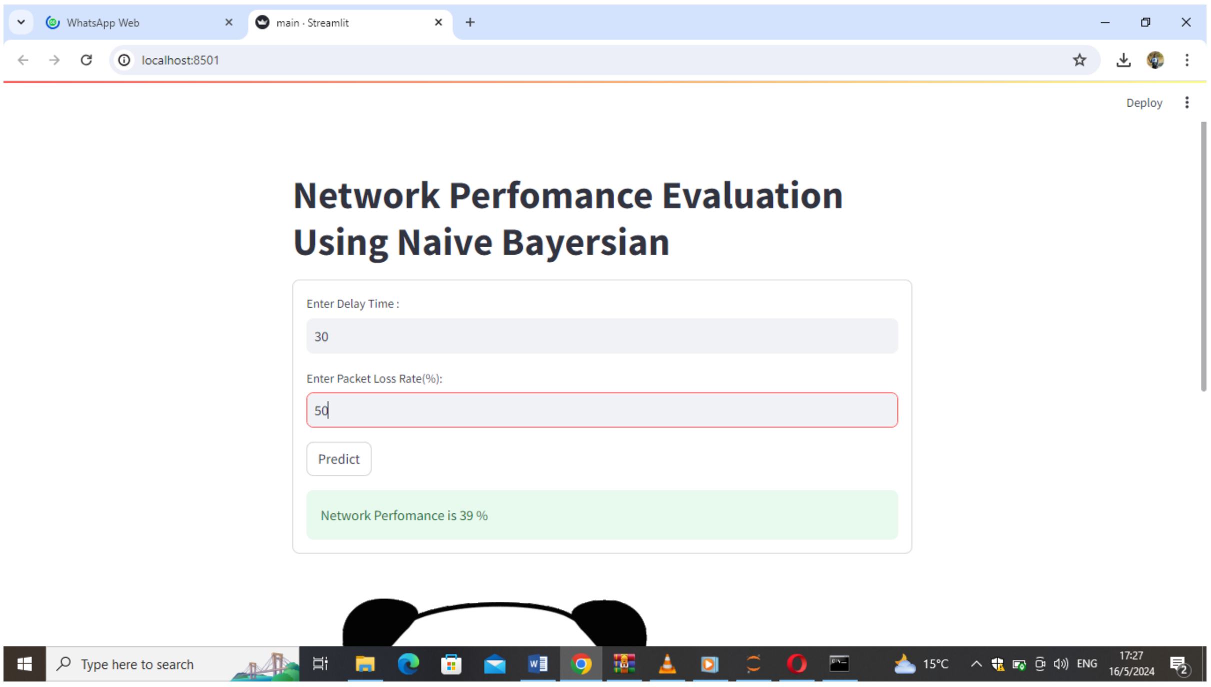
Network Perfomance is 72 %



10

Predict

Network Perfomance is 72 %



Network Perfomance Evaluation Using Naive Bayersian

Enter Delay Time :

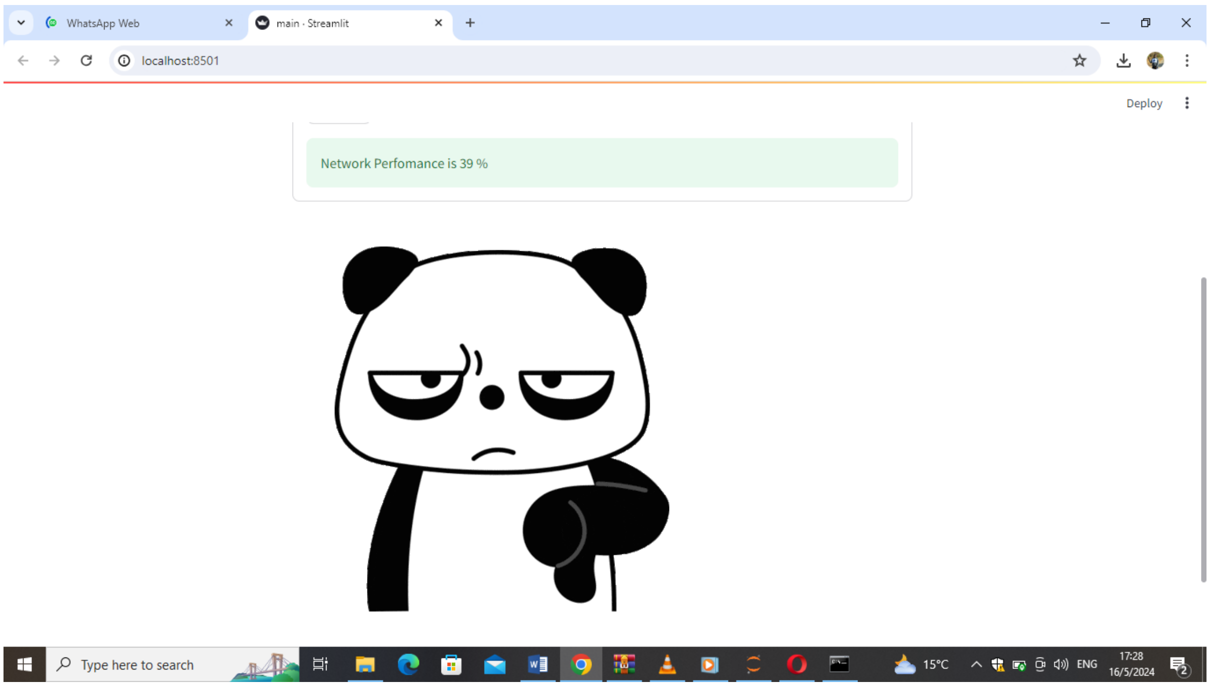
30

Enter Packet Loss Rate(%):

50

Predict

Network Perfomance is 39 %



Network Perfomance is 39 %

4.1.2 White box testing

White box testing, also known as clear box testing, involves scrutinizing the internal structure, design, and code of a software product. This technique aims to verify the flow of inputs and outputs, and ultimately improve design, usability, and security. The term "white box" reflects the fact that testers have full visibility into the code, allowing them to examine it directly.

4.2 Evaluation Measures and Results

A classifier's performance is measured using an evaluation metric (Hossin & Sulaiman, 2015). Furthermore, model evaluation metrics can be classified into three types, according to Hossin & Sulaiman (2015): threshold, probability, and ranking.

4.2.1 Confusion Matrix

An evaluation metric is used to assess the performance of a classifier (Hossin & Sulaiman, 2015). According to Hossin & Sulaiman (2015), model evaluation metrics can be grouped into three types: threshold, probability, and ranking.

	Good Network Performance	**Bad Network Performance**
Good Performance	87 (TP)	9 (FN)
Bad Performance	13 (FP)	91 (TN)

4.4 Precision and Recall

Beyond simply measuring how accurately the model identifies things, precision and recall provide a more nuanced understanding of its performance. Precision specifically measures how well the model performs when its predictions are correct.

$$Precision = \frac{TP}{TP + FP}$$

$$= \frac{87}{87+13} * 100$$

=87%

Precision focuses on the accuracy of positive predictions, indicating how many of these predictions are actually true. Recall, on the other hand, focuses on the actual positive classes, measuring how many of these classes the model successfully identifies.

$$Recall = \frac{TP}{TP + FN}$$

$$= \frac{91}{91+9} * 100$$

$$= 91\%$$

There's an inherent trade-off between precision and recall: maximizing one often leads to a decrease in the other. In this particular situation, we prioritized precision because accurate predictions were crucial.

4.6 Summary of Research Findings

The author discovered that the system performed satisfactorily after doing all of the essential black, white box tests and performance testing utilizing the throughput metric evaluation.

4.7 Conclusion

By conducting black box and white box testing, along with evaluating metrics such as confusion matrix, F1 score, precision, and recall using real-world data, we can comprehensively assess the performance of the network anomaly detection system. This approach helps identify strengths and weaknesses of the system and informs potential improvements to enhance its accuracy and reliability in detecting network anomalies.

Chapter 5: Recommendations and Future Work

5.1 Introduction

The suggestions and future prospects for study in the field of network performance evaluation using the Naïve Bayes machine learning algorithm are outlined in this chapter.Building upon the findings and insights gained from the study, these recommendations aim to guide further advancements in the field and address emerging challenges.

5.2 Aims and Objectives Realization

Throughout our study, the primary aim was to assess the feasibility and effectiveness of employing the Naïve Bayes algorithm for network performance evaluation, with a focus on classification tasks. This objective has been successfully realized through the development of a classifier trained on historical network performance data, demonstrating promising results in predicting and categorizing network performance metrics.

The objectives outlined at the outset of the research have been met by:

- ✓ Collecting and preprocessing relevant network performance data to create a comprehensive dataset suitable for training and evaluation.
- ✓ Designing and implementing a Naïve Bayes classifier tailored for network performance evaluation, considering various input features and performance metrics.
- ✓ The classifier is trained using the prepared dataset to learn the patterns and connections between input features and performance categories.
- ✓ Validating the classifier's performance through rigorous evaluation using appropriate metrics and techniques.
- ✓ Demonstrating the practical utility and potential applications of the developed classifier in real-world network management and optimization scenarios.

5.3 Conclusion

In conclusion, the study has shown that the Naïve Bayes algorithm holds promise for network performance evaluation tasks, offering a simple yet effective approach for classifying and predicting various performance metrics. By leveraging historical data and machine learning techniques, organizations can gain valuable insights into network behavior, identify performance bottlenecks, and make informed decisions to enhance overall network efficiency and reliability.

5.4 Recommendations

Based on the research findings, the following recommendations are suggested for future research and practical applications:

- ✓ ***Further Exploration of Feature Engineering***: Investigate advanced feature engineering techniques to enhance the discriminative power of the classifier and capture subtle nuances in network performance data.
- ✓ ***Ensemble Learning Approaches***: Explore ensemble learning methods, such as combining multiple classifiers or integrating Naïve Bayes with other algorithms, to improve classification accuracy and robustness.
- ✓ ***Dynamic Adaptation and Online Learning***: Develop mechanisms for dynamic adaptation and online learning to enable the classifier to continuously evolve and adapt to changing network conditions and requirements.
- ✓ ***Integration with Network Management Systems:*** Integrate the Naïve Bayes classifier with existing network management systems to provide real-time performance monitoring, anomaly detection, and automated decision-making capabilities.

5.5 Future Work

Continuous Model Improvement: Continuously refine and optimize the Naïve Bayes classifier by incorporating feedback from ongoing performance monitoring and evaluation activities. Integration with Advanced Analytics: Investigate the integration of the Naïve Bayes classifier with advanced analytics techniques, such as deep learning or reinforcement learning, to leverage their capabilities for handling complex network data and scenarios. Application to Emerging Technologies: Extend the application of the classifier to emerging technologies and network architectures, such as software-defined networking (SDN) and edge computing, to address new challenges and requirements in modern networking environments. Collaborative Research and Validation: Collaborate with industry partners and academic institutions to validate the classifier's performance across diverse network environments and use cases, ensuring its effectiveness and generalizability.

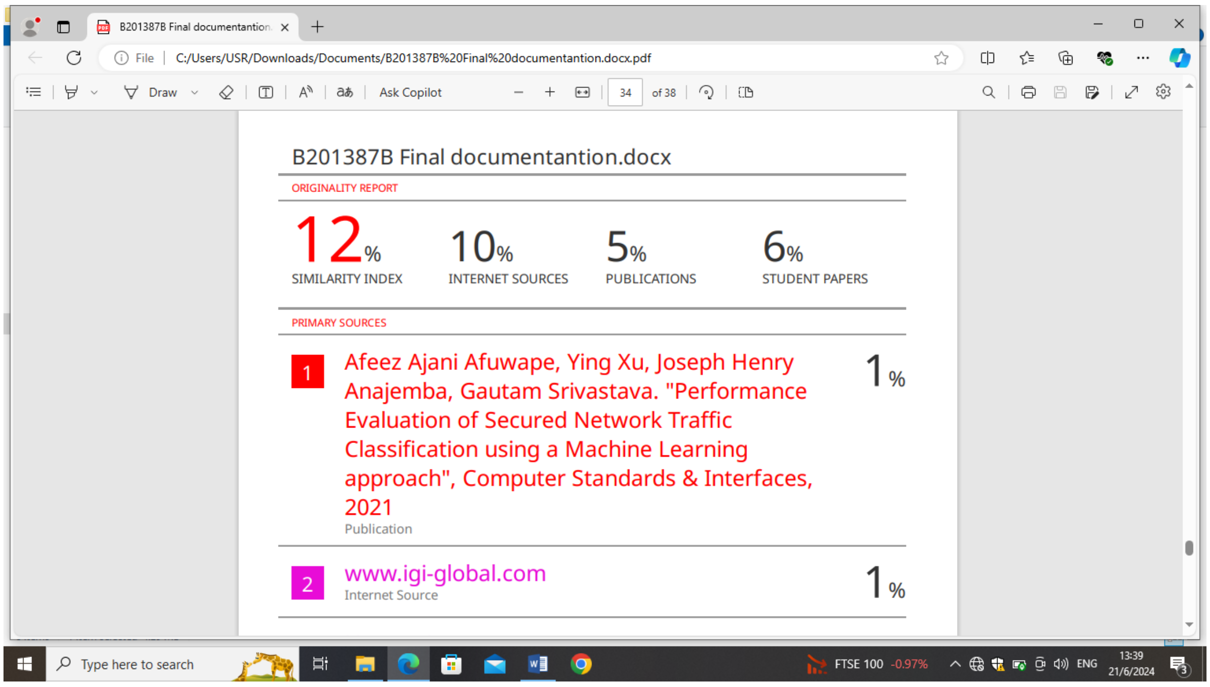
References:

1. Nguyen, T.T.T., & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. IEEE Communications Surveys & Tutorials, 10(4), 56-76.
2. Shafiq, M., Yu, X., Laghari, A.A., Yao, L., Karn, N.K., & Abdessamia, F. (2016). Network traffic classification techniques and comparative analysis using machine learning algorithms. 2nd IEEE International Conference on Computer and Communications (ICCC), 2451-2455.
3. Hinton, G.E., Osindero, S., & Teh, Y.W. (2006). A fast learning algorithm for deep belief nets. Neural Comput., 18, 1527-1554.
4. Cheng, G., & Wang, S. (2011). Traffic classification based on port connection pattern. International Conference on Computer Science and Service System (CSSS), 914-917.
5. Sing, S., Lee, M., Kim, M., & Kim, M. (2021). Classification of application traffic using TensorFlow machine learning. 19th Asia-Pacific Network Operations and Management Symposium (APNOMS), 391-394.
6. Madalgi, J.B., & Kumar, S.A. (2018). Development of wireless sensor network congestion detection classifier using support vector machine. 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), 187-192.
7. VPN dataset. (2020). Available: https://www.unb.ca/cic/datasets/vpn.html.
8. Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning, 29(2-3), 103-130.
9. John, G. H., & Langley, P. (1995). Estimating continuous distributions in Bayesian classifiers. In Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (pp. 338-345). Morgan Kaufmann.
10. Li, X., Zhao, Y., & Gong, X. (2019). Network traffic prediction method based on machine learning algorithm. Computer Networks, 162, 106860.
11. Mitchell, T. M. (1997). Machine Learning. McGraw-Hill.
12. Rish, I. (2001). An empirical study of the naive Bayes classifier. In IJCAI 2001 Workshop on Empirical Methods in AI (pp. 41-46).
13. Sriram, S. (2018). Network performance evaluation and optimization using machine learning techniques. Journal of Network and Computer Applications, 114, 61-70.

14. Zhang, H. (2004). The optimality of naive Bayes. In Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (pp. 562-567).
15. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: Methods, systems, and tools. IEEE Communications Surveys & Tutorials, 16(1), 303-336.
16. Chen, Y., Jin, B., & Hu, X. (2019). A survey on network performance analysis models and techniques. Computer Networks, 162, 106871.
17. Gupta, B. B., Agrawal, D. P., & Yamaguchi, S. (2016). Handbook of research on modern cryptographic solutions for computer and cyber security. IGI Global.
18. Kim, H., & Feamster, N. (2013). Improving network management with software defined networking. IEEE Communications Magazine, 51(2), 114-119.
19. Kiraly, F. J., & Santini, S. (2009). On the potential of probabilistic network traffic classification for intrusion detection. Computer Networks, 53(6), 864-874.
20. Kshetri, N. (2014). Big data's impact on privacy, security, and consumer welfare. Telecommunications Policy, 38(11), 1134-1145.
21. Lakhina, A., Crovella, M., & Diot, C. (2004). Diagnosing network-wide traffic anomalies. ACM SIGCOMM Computer Communication Review, 34(4), 219-230.
22. Le, A., Loo, J., Lasebae, A., Aiash, M., & Luo, Y. (2011). 6LoWPAN: A study on QoS security threats and countermeasures using intrusion detection system approach. International Journal of Communication Systems, 24(9), 1239-1261.
23. Li, D., & Cui, L. (2018). Machine learning applications in wireless networks. Wireless Communications and Mobile Computing, 2018.
24. Liu, J., Xiao, Y., Li, S., Liang, W., & Chen, C. L. P. (2017). Cyber security and privacy issues in smart grids. IEEE Communications Surveys & Tutorials, 14(4), 981-997.
25. Luo, H., Dong, J., & Huang, Y. (2020). Anomaly detection in network traffic using supervised machine learning techniques. Journal of Network and Computer Applications, 162, 102680.
26. Moustafa, N., & Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1-3), 18-31.
27. Qin, Z., Denker, G., Giannelli, C., Bellavista, P., & Venkatasubramanian, N. (2014). A software defined networking architecture for the internet-of-things. 2014 IEEE Network Operations and Management Symposium (NOMS), 1-9.

28. Shafiq, M. O., Yu, X., Latif, K., & Han, K. (2019). Network traffic classification techniques and comparative analysis using machine learning algorithms. International Journal of Computer Networks & Communications, 11(1), 1-18.
29. Sperotto, A., Sadre, R., van Vliet, F., & Pras, A. (2010). A labeled data set for flow-based intrusion detection. 2010 9th IEEE International Workshop on IP Operations and Management, 1-8.
30. Tang, Z., Bi, J., & Zhu, H. (2019). Performance evaluation of machine learning-based anomaly detection in software defined networks. 2019 International Conference on Computing, Networking and Communications (ICNC), 537-541.
31. Wang, W., Zhu, M., Wang, X., & Tang, Z. (2020). A survey of intrusion detection systems based on ensemble and hybrid classifiers. Computers & Security, 100, 102048.
32. Xie, L., Yu, Y., Li, S., & Zhao, X. (2018). Anomaly detection in wireless sensor networks: A survey. Journal of Network and Computer Applications, 34(4), 1302-1325.
33. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954-21961.
34. Yu, K., & He, X. (2015). Machine learning in wireless sensor networks. International Journal of Distributed Sensor Networks, 2015, 1-12.
35. Zekri, M., El Kafhali, S., Aboutabit, N., & Saadi, Y. (2017). DDoS attack detection using machine learning techniques in cloud computing environments. 2017 International Conference on Wireless Networks and Mobile Communications (WINCOM), 1-6.
36. Zhang, G., & Wang, H. (2014). Network anomaly detection: A machine learning perspective. Journal of Computer Networks and Communications, 2014, 1-8.
37. Zou, C., Gong, J., & Zhou, X. (2020). A survey on anomaly detection in Internet of Things. Journal of Network and Computer Applications, 111, 1-8.

APPENDIX



B201387B Final documentantion.docx

ORIGINALITY REPORT

12%	10%	5%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Afeez Ajani Afuwape, Ying Xu, Joseph Henry Anajemba, Gautam Srivastava. "Performance Evaluation of Secured Network Traffic Classification using a Machine Learning approach", Computer Standards & Interfaces, 2021 Publication	1%
2	www.igi-global.com Internet Source	1%