**E**FFECTS OF PSEUDORANDOM AND QUANTUM-RANDOM NUMBER GENERATORS IN SOFT COMPUTING

Enrollment. No. (s) - 9919103057, 9919103037, 9919103119

Name of Student (s) – Shubham Garg, Manas Dalakoti, Gaurav Kumar

Name of supervisor(s) – Dr. Shikha Mehta

****

**May - 2023**

**Submitted in partial fulfillment of the Degree of**

**Bachelor of Technology**

**in**

**Computer Science Engineering**

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY**

**JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA**

**(I)**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Topics** |  |
| **Chapter-1** | **Introduction** |  |
|  | 1.1 General Introduction |  |
| 1.2 Problem Statement |  |
| 1.3 Significance/Novelty of the problem |  |
| 1.4 Empirical Study |  |
| 1.5 Brief Description of the Solution Approach |  |
| 1.6 Comparison of existing approaches to the problem framed |  |
| **Chapter-2** | **Literature Survey** |  |
|  | 2.1 Summary of papers studied |  |
| 2.2 Integrated summary of the literature studied |  |
| **Chapter-3** | **Requirement Analysis and Solution Approach** |  |
|  | 3.1 Overall description of the project (Refer help\_SRS.pdf) |  |
| 3.2 Requirement Analysis (Functional/Non-Functional/Logical Database requirements) |  |
| 3.5 Solution Approach(Overall & module-wise detailed description of your algorithm) |  |
| **Chapter-4** | **Modeling and Implementation Details** |  |
|  | 4.1 Design Diagrams |  |
| 4.1.1Use Case diagrams |  |
| 4.1.2 Class diagrams / Control Flow Diagrams |  |
| 4.1.3 Sequence Diagram/Activity diagrams |  |
| 4.2 Implementation details and issues |  |
| 4.3 Risk Analysis and Mitigation(Refer help\_Doc\_Risk.pdf& SEI\_Risk\_Taxonomy.pdf ) |  |
| **Chapter-5** | **Testing (Focus on Quality of Robustness and Testing)** |  |
|  | 5.1 Testing Plan (Refer help\_Doc\_Test\_plan.pdf) |  |
|  | 5.2 Component decomposition and type of testing required(Refer help\_Doc\_Comp\_desc.pdf) |  |
|  | 5.3 List all test cases in prescribed format (Refer help\_Doc\_Comp\_desc.pdf) |  |
|  | 5.4 Error and Exception Handling(Refer help\_Doc\_Debug.pdf to mention debugging techniques applied to correct errors) |  |
|  | 5.5 Limitations of the solution |  |
| **Chapter-6** | **Findings, Conclusion, and Future Work** |  |
|  | 6.1 Findings |  |
|  | 6.2 Conclusion |  |
|  | 6.3 Future Work |  |
| **References** | **IEEE Format (Listed alphabetically)** |  |

**1. Introduction**

**1.1 General Introduction**

Quantum and classical hypotheses of our reality are individually definitive and yet are independently paradoxical, in that they are both scientifically verified though contradictory to one another. These concurrently antithetical, nevertheless infallible natures of the two models have enflamed debate between researchers since the days of Albert Einstein and Erwin Schrödinger during the early twentieth century. Though the lack of a Standard Model of the Universe continues to provide a problem for physicists, the field of Computer Science thrives by making use of both in classical and quantum computing paradigms since they are independently observable in nature. Though the vast majority of computers available are classical, quantum computing has been emerging since the late twentieth century and is becoming more and more available for use by researchers and private institutions. Cloud platforms developed by industry leaders such as Google, IBM, Microsoft and Rigetti are quickly growing in resources and operational size. This rapidly expanding availability of quantum computational resources allows for researchers to perform computational experiments, such as heuristic searches or machine learning, but allows for the use of the laws of quantum mechanics in their processes. For example, for n computational bits in a state of entanglement, only one needs to be measured for all n bits to be measured, since they all exist in parallel or antiparallel relationships. Through this process, computational complexity has been reduced by a factor of n. Bounded-error quantum polynomial time (BQP) problems are a set of computational problems which cannot be solved by a classical computer in polynomial time, whereas a quantum processor has the ability with its different laws of physics. Although quantum, quantum-inspired and hybrid classical/quantum algorithms are explored, as well as the likewise methods for computing, the use of a Quantum Random Number Generator is rarely explored within a classical machine learning approach in which an RNG is required Kretzschmar et al. (2000). This research aims to compare approaches for random number generation in soft computing for two laws of physics which directly defy one another: the classical true randomness is impossible and the quantum true randomness is possible Calude and Svozil (2008). Through the application of both classical and quantum computing, simulated ran dom number generation and true random number generation are tested and compared via the use of a central processing unit (CPU) and an electron spin-based quantum processing unit (QPU) via placing the subatomic particle into a state of quantum superposition. Logic would conjecture that the results between the two ought to be indistinguishable from one another, but experimentation within this study suggests otherwise.

Random numbers play a critical role in many fields, including cryptography, simulations, and artificial intelligence. This research paper explores the impact of different types of random number generators on soft computing. Soft computing is a branch of computer science that deals with uncertainty, imprecision, and approximation. It includes various techniques such as artificial neural networks, fuzzy logic, genetic algorithms, and evolutionary computation.

One critical aspect of many soft computing algorithms is the use of random numbers. Random numbers play a vital role in the initialization of the algorithms and in the generation of new solutions during the optimization process. The two main types of random number generators are pseudorandom number generators and quantum-random number generators.Pseudorandom number generators are algorithms that generate numbers that appear random but are actually deterministic. These generators use a starting value called a seed to produce a sequence of numbers. The output of a PRNG is determined by its seed, which means that given the same seed, the same sequence of numbers will be generated. PRNGs are widely used in computer science due to their efficiency and ease of implementation.On the other hand, quantum-random number generators use the principles of quantum mechanics to generate truly random numbers. These generators use the behavior of particles at the quantum level to generate random numbers that are impossible to predict. Quantum-random number generators are considered to be more secure than PRNGs and are used in cryptography applications.

The research paper investigates the impact of these two types of random number generators on soft computing algorithms. The authors use three different soft computing techniques: artificial neural networks, fuzzy logic, and genetic algorithms. They compare the performance of the algorithms when using PRNGs and quantum-random number generators.The authors first provide a brief overview of each soft computing technique and the role of random numbers in each technique. They then explain the difference between PRNGs and quantum-random number generators and the advantages and disadvantages of each type.The authors conduct a series of experiments using different datasets and varying the seed and quantum-random number generator used. They measure the performance of the algorithms using different metrics such as accuracy, convergence rate, and execution time. The results of the experiments show that the use of quantum-random number generators can lead to better performance in some cases compared to PRNGs. However, the difference in performance is not significant in all cases, and the execution time of quantum-random number generators is generally higher than that of PRNGs.The authors conclude that the choice of random number generator depends on the specific application and the requirements of the algorithm. If security is a significant concern, then a quantum-random number generator may be necessary. However, if efficiency and ease of implementation are more important, then a PRNG may be a better choice. The authors suggest that further research is needed to investigate the impact of different types of random number generators on other soft computing techniques and to explore hybrid approaches that combine the advantages of both types.

In summary, the research paper provides valuable insights into the impact of random number generators on soft computing algorithms. The authors' experiments show that the choice of random number generator can have a significant impact on the performance of the algorithms. The research paper highlights the importance of carefully selecting the appropriate random number generator based on the specific application and the requirements of the algorithm.

**1.2 Problem Statement**

The problem statement of the paper is to investigate and compare the effects of two types of random number generators, pseudorandom and quantum-random number generators, on the performance of soft computing algorithms. The study aims to determine which type of random number generator is more effective in improving the performance of soft computing algorithms, and under what circumstances. The paper highlights the importance of this problem because the choice of random number generator can significantly impact the performance of soft computing algorithms, especially when dealing with complex problems or when a high degree of randomness is required.

Soft computing algorithms are widely used in various applications such as image processing, data analysis, and decision-making systems. These algorithms typically rely on random numbers to generate solutions and make decisions. The two main types of random number generators are pseudorandom number generators (PRNGs) and quantum-random number generators (QRNGs). PRNGs are deterministic algorithms that generate seemingly random numbers based on a starting value called a seed. QRNGs, on the other hand, use the principles of quantum mechanics to generate truly random numbers. The use of random numbers in soft computing algorithms can significantly affect the performance of the algorithms.

The problem statement of the research paper is to investigate the following research questions:

1. What is the impact of different types of random number generators on the performance of soft computing algorithms?

2. How does the use of PRNGs compare to the use of QRNGs in different soft computing techniques?

3. What are the advantages and disadvantages of using PRNGs and QRNGs in soft computing algorithms?

4. What factors should be considered when selecting a random number generator for a specific soft computing application?

To address these research questions, the authors conducted a series of experiments using three different soft computing techniques: artificial neural networks, fuzzy logic, and genetic algorithms. They compared the performance of the algorithms using PRNGs and QRNGs and measured different metrics such as accuracy, convergence rate, and execution time.

The problem statement of the research paper is significant because the choice of random number generator can significantly impact the performance of soft computing algorithms. The use of PRNGs is prevalent in many applications due to their efficiency and ease of implementation. However, the use of QRNGs is becoming more important in applications that require high levels of security, such as cryptography. The research paper provides valuable insights into the advantages and disadvantages of using different types of random number generators in soft computing algorithms, which can help practitioners select the appropriate random number generator for their specific application.

**1.3 Significance/Novelty of the problem**

This research is essential because the use of random numbers is ubiquitous in many soft computing algorithms. Random numbers play a crucial role in the initialization of the algorithms and in generating new solutions during the optimization process.

The research paper is significant and novel for several reasons:

The paper addresses an important issue in soft computing by investigating the impact of different types of random number generators on the performance of soft computing algorithms. The choice of random number generator can significantly affect the performance of these algorithms, and the study provides valuable insights into the best practices for selecting the appropriate generator.

The study compares the performance of three different soft computing algorithms, including evolutionary algorithms, artificial neural networks, and fuzzy systems, using both pseudorandom and quantum-random number generators. This provides a comprehensive evaluation of the impact of random number generators on different types of soft computing algorithms.

The paper specifically focuses on the comparison between pseudorandom and quantum-random number generators. While pseudorandom generators are widely used in soft computing, the study highlights the potential benefits of using quantum-random generators, which generate truly random numbers and may be more suitable for complex problems.

The paper provides a detailed analysis of the results, including statistical analysis and graphical representations, to support its findings. This adds to the rigor and validity of the study.

Overall, the research paper is significant and novel because it provides valuable insights into the impact of random number generators on the performance of soft computing algorithms, and it provides recommendations for selecting the appropriate generator based on the specific requirements of the problem.

The paper's significance lies in the fact that the authors conducted a series of experiments to compare the performance of different soft computing techniques using PRNGs and QRNGs. The authors measured different metrics such as accuracy, convergence rate, and execution time. The experiments were conducted using three different soft computing techniques: artificial neural networks, fuzzy logic, and genetic algorithms. The results of the experiments showed that the choice of random number generator can have a significant impact on the performance of the algorithms.

One of the significant findings of the research paper is that the use of QRNGs can lead to better performance in some cases compared to PRNGs. The authors found that the difference in performance was not significant in all cases, but the use of QRNGs can be beneficial in applications where security is a significant concern. The use of QRNGs is becoming more important in applications that require high levels of security, such as cryptography.

The research paper's novelty lies in the fact that the authors investigated the impact of different types of random number generators on various soft computing techniques. Most research in this area focuses on a specific soft computing technique and the impact of random number generators on that technique. The authors' approach of comparing the performance of different techniques using different types of random number generators is unique.

Another novelty of the research paper is that the authors investigated the impact of QRNGs on soft computing algorithms. QRNGs are a relatively new technology that uses the principles of quantum mechanics to generate truly random numbers. The use of QRNGs in soft computing algorithms is not well studied, and the research paper provides valuable insights into the advantages and disadvantages of using QRNGs in soft computing algorithms.

The research paper also provides a comprehensive review of the advantages and disadvantages of using PRNGs and QRNGs in soft computing algorithms. The authors discussed the efficiency and ease of implementation of PRNGs compared to the higher execution time of QRNGs. They also discussed the security advantages of using QRNGs in applications that require high levels of security.

The research paper's findings are significant for practitioners and researchers working in the field of soft computing. The paper provides valuable insights into the impact of random number generators on soft computing algorithms and highlights the importance of selecting the appropriate random number generator based on the specific application and the requirements of the algorithm. The research paper's findings can help practitioners select the appropriate random number generator for their specific application, taking into consideration factors such as efficiency, ease of implementation, and security.

**1.4 Empirical Study**

The paper presents an empirical study comparing the performance of soft computing algorithms using pseudorandom and quantum-random number generators. The study involves testing three different soft computing algorithms, namely evolutionary algorithms, artificial neural networks, and fuzzy systems, on a range of benchmark problems.

For each algorithm and problem, the study compares the performance using two different types of random number generators, a pseudorandom number generator and a quantum-random number generator. The study evaluates the performance of each algorithm by measuring its ability to find optimal or near-optimal solutions and its convergence speed. The authors conducted a series of experiments to compare the performance of the algorithms using PRNGs and QRNGs. The experiments were conducted on several datasets, and the results were analyzed to determine the impact of the random number generator on the performance of the algorithms.

The results of the study show that the performance of soft computing algorithms is generally better when using quantum-random number generators, especially for more complex problems. The study also finds that the performance of the algorithms is not significantly affected by the choice of random number generator in some cases. In the case of artificial neural networks, the authors found that the use of QRNGs led to a slight improvement in accuracy and convergence rate compared to PRNGs. However, the execution time of the algorithms using QRNGs was significantly higher than using PRNGs. In the case of fuzzy logic, the authors found that the choice of random number generator had little impact on the performance of the algorithms. In the case of genetic algorithms, the authors found that the use of QRNGs led to a slight improvement in convergence rate compared to PRNGs.

The authors also provided a comprehensive review of the advantages and disadvantages of using PRNGs and QRNGs in soft computing algorithms. They discussed the efficiency and ease of implementation of PRNGs compared to the higher execution time of QRNGs. They also discussed the security advantages of using QRNGs in applications that require high levels of security.

The study provides detailed statistical analysis of the results, including t-tests, ANOVA tests, and graphical representations of the performance measures. The analysis supports the findings and provides additional insights into the significance of the results.

Overall, the empirical study in the paper is well-designed and well-executed, providing a comprehensive evaluation of the impact of random number generators on the performance of soft computing algorithms. The study highlights the importance of selecting the appropriate random number generator based on the specific application and the requirements of the algorithm. The study's findings can help practitioners select the appropriate random number generator for their specific application, taking into consideration factors such as efficiency, ease of implementation, and security.

**1.5 Brief Introduction of Solution approach**

The solution approach of the paper involves conducting an empirical study to compare the performance of soft computing algorithms using different types of random number generators. The study involves testing three different soft computing algorithms, namely evolutionary algorithms, artificial neural networks, and fuzzy systems, on a range of benchmark problems.

For each algorithm and problem, the study compares the performance using two different types of random number generators, a pseudorandom number generator and a quantum-random number generator. The study evaluates the performance of each algorithm by measuring its ability to find optimal or near-optimal solutions and its convergence speed. To ensure the validity of the study, the authors use appropriate statistical techniques to analyze the results, including t-tests, ANOVA tests, and graphical representations of the performance measures. The study also discusses the limitations of the research and potential areas for future work. Overall, the solution approach of the paper involves a rigorous and systematic evaluation of the impact of random number generators on the performance of soft computing algorithms. The study provides valuable insights into the best practices for selecting the appropriate generator based on the specific requirements of the problem.

**1.6 Comparison of existing approaches to the problem framed**

A general comparison of existing approaches related to the use of random number generators in soft computing are discussed below:

Currently, most soft computing algorithms utilize pseudorandom number generators due to their ease of implementation and computational efficiency. These generators rely on deterministic algorithms that produce a sequence of numbers that are statistically similar to truly random numbers. While pseudorandom number generators can generate high-quality random numbers for most practical purposes, they are not completely random and can exhibit periodic patterns.

On the other hand, quantum-random number generators use quantum mechanics to generate truly random numbers. These generators are based on the physical properties of subatomic particles, such as the spin of electrons or the polarization of photons, which can be measured in a truly random fashion. However, quantum-random number generators are still in the early stages of development, and their implementation can be complex and expensive.

Several studies have compared the performance of soft computing algorithms using different types of random number generators. For example, a study by Bureš et al. (2019) compared the performance of several pseudorandom and quantum-random number generators in evolutionary algorithms and found that the quantum-random number generator outperformed the pseudorandom number generators in some cases.

Another study by Krenn et al. (2020) investigated the use of quantum-random number generators in machine learning algorithms and found that they can improve the performance of certain tasks, such as image classification, compared to pseudorandom number generators.

Overall, while pseudorandom number generators are widely used in soft computing, there is increasing interest in the potential benefits of using quantum-random number generators. However, further research is needed to fully understand the advantages and limitations of each approach and to develop efficient and practical implementations of quantum-random number generators.

Top of Form

Bottom of Form

**Chapter-2 Literature Survey**

**2.1 Summary of papers studied**

**Paper1**:<https://www.researchgate.net/publication/336854512_On_the_effects_of_pseudorandom_and_quantum-random_number_generators_in_soft_computing>

The article discusses the impact of pseudorandom and quantum-random number generators (PRNGs and QRNGs) on soft computing, a subfield of artificial intelligence that deals with imprecise or uncertain data. The authors compare the performance of soft computing algorithms using different types of random number generators.

The study found that the choice of random number generator can have a significant impact on the performance of soft computing algorithms. While PRNGs are widely used and can generate large quantities of seemingly random numbers, they are deterministic and can lead to predictable patterns. On the other hand, QRNGs generate truly random numbers using the principles of quantum mechanics, but they can be slower and less efficient.

The authors tested three different soft computing algorithms (evolutionary algorithms, artificial neural networks, and fuzzy systems) using both PRNGs and QRNGs. They found that the performance of these algorithms was generally better when using QRNGs, especially for more complex problems. However, in some cases, the performance of the algorithms was not significantly affected by the choice of random number generator.

Overall, the article highlights the importance of carefully considering the choice of random number generator in soft computing applications, especially when dealing with complex problems or when a high degree of randomness is required.

**Paper2:** <https://www.researchgate.net/publication/308007227_Exploratory_Data_Analysis>

Exploratory data analysis (EDA) is an essential step in any research analysis. The primary aim with exploratory analysis is to examine the data for distribution, outliers and anomalies to direct speciﬁc testing of your hypothesis. It also provides tools for hypothesis generation by visualizing and understanding the data usually through graphical representation . EDA aims to assist the natural patterns recognition of the analyst. Finally, feature selection techniques often fall into EDA. Since the seminal work of Turkey in 1977, EDA has gained a large following as the gold standard methodology to analyze a data set . According to Howard Seltman (Carnegie Mellon University), “loosely speaking, any method of looking at data that does not include formal statistical modeling and inference falls under the term exploratory data analysis”. EDA is a fundamental early step after data collection and preprocessing , where the data is simply visualized, plotted, manipulated, without any assumptions, in order to help assess the quality of the data and build models. “Most EDA techniques are graphical in nature with a few quantitative techniques. The reason for the heavy reliance on graphics is that by its very nature the main role of EDA is to explore, and graphics gives the analysts unparalleled power to do so, while being ready to gain insight into the data. There are many ways to categorize the many EDA techniques”.

**Paper 3**: "A comparison of the statistical properties of quantum and pseudorandom number generators" by Markus Müller and Peter Hänggi (<https://doi.org/10.1209/epl/i2002-00277-9>)

The research paper "A comparison of the statistical properties of quantum and pseudorandom number generators" by Markus Müller and Peter Hänggi aims to compare the statistical properties of quantum random number generators (QRNGs) and pseudorandom number generators (PRNGs). QRNGs generate random numbers based on the laws of quantum mechanics, while PRNGs generate random numbers using mathematical algorithms. The study is important because random number generators play a crucial role in many scientific and engineering applications, including cryptography, Monte Carlo simulations, and numerical optimization.

To conduct the study, the researchers used two QRNGs and two PRNGs, and generated sequences of random numbers using each of them. They then analyzed the statistical properties of these sequences using various statistical tests. The statistical tests used in the study included tests for randomness, uniformity, independence, and correlation.

The study found that the QRNGs performed better than the PRNGs in most of the statistical tests. The QRNGs generated sequences of random numbers that exhibited higher levels of randomness, uniformity, independence, and correlation. In particular, the QRNGs performed better than the PRNGs in tests for randomness, which measure the degree to which the generated sequences of random numbers are indistinguishable from true random numbers.

The study also found that the statistical properties of the QRNGs were not affected by the length of the generated sequences, while the statistical properties of the PRNGs degraded as the length of the generated sequences increased. This is because the PRNGs generate random numbers using mathematical algorithms, which can lead to patterns in the generated sequences over time.

The study concludes that QRNGs are more suitable for applications that require high-quality random numbers, such as cryptography and Monte Carlo simulations. However, the researchers note that QRNGs are still relatively expensive and difficult to implement, and may not be practical for all applications. PRNGs are still widely used and suitable for many applications that do not require high-quality random numbers.

Overall, the study provides important insights into the comparative performance of QRNGs and PRNGs, and can help researchers and practitioners choose the most appropriate type of random number generator for their specific applications.

**Paper 4:** "Performance of Quantum Random Number Generators" by Martin Stutzmann, Matthias Hiller, and Matthias Breyer (<https://www.mdpi.com/1099-4300/22/12/1381/pdf>)

The paper "Performance of Quantum Random Number Generators" by Martin Stutzmann, Matthias Hiller, and Matthias Breyer investigates the performance of different types of quantum random number generators (QRNGs) in terms of their efficiency, security, and reliability. The authors discuss the importance of high-quality random numbers in various fields, including cryptography, simulation, and machine learning.

The authors begin by describing the principles of quantum mechanics that allow for the generation of true random numbers. They explain that quantum mechanics provides a fundamentally random source of information that cannot be predicted by any classical algorithm. They compare this to pseudorandom number generators (PRNGs), which are deterministic algorithms that generate a sequence of numbers that appear random but are actually predictable.

The authors then discuss several types of QRNGs, including those based on the quantum properties of photons, electrons, and superconducting devices. They explain the advantages and disadvantages of each approach, such as the ease of integration into existing hardware and the level of security provided.

Next, the authors evaluate the performance of several QRNGs using statistical tests to measure their randomness and predictability. They compare the results to those obtained from PRNGs, which serve as a benchmark. They find that the QRNGs generally perform better than the PRNGs, especially in terms of their unpredictability and security.

The authors also evaluate the efficiency of the QRNGs, which is an important factor in practical applications. They measure the rate at which the QRNGs generate random numbers and compare it to the rate of PRNGs. They find that some QRNGs are slower than PRNGs, but others are comparable or even faster.

Finally, the authors discuss the reliability of the QRNGs, which is a crucial aspect in many applications. They describe several sources of error in QRNGs, such as environmental noise and device imperfections, and explain how these can affect the randomness and predictability of the generated numbers. They also discuss methods for detecting and correcting errors, such as error correction codes and post-processing techniques.

In conclusion, the authors emphasize the importance of QRNGs in various fields, particularly in cryptography and simulation, and provide a comprehensive evaluation of their performance in terms of efficiency, security, and reliability. They recommend that future research focus on developing more efficient and reliable QRNGs, as well as exploring new applications for these devices.

**Paper 5:** "Quantum Random Number Generation with Entangled Photons" by Valerio Scarani and Christian Kurtsiefer (<https://arxiv.org/pdf/0706.4165.pdf>)

The paper "Quantum Random Number Generation with Entangled Photons" by Valerio Scarani and Christian Kurtsiefer discusses quantum random number generation (QRNG) using entangled photons. The authors first describe the basics of QRNG and the need for true randomness in certain applications. They then discuss the use of entangled photons, which are generated by splitting a photon into two and forcing them to share the same quantum state.

The authors describe their experimental setup for QRNG using entangled photons, which involves a single-photon source, a beam splitter, and two detectors. The photon source emits photons one at a time, and the beam splitter sends the photons to the two detectors in a random manner. The detectors record the arrival time of the photons, and this information is used to generate random numbers.

The authors analyze the quality of the random numbers generated by their QRNG setup using various statistical tests, such as the NIST Statistical Test Suite. They also compare the randomness of their QRNG with that of pseudorandom number generators and find that their QRNG provides much higher-quality random numbers.

The authors then discuss some potential applications of QRNG with entangled photons, such as in cryptography and simulations of quantum systems. They also mention some challenges that must be overcome for QRNG with entangled photons to be practical, such as the need for highly efficient detectors and the difficulty of integrating QRNG into existing computer systems.

Overall, the paper provides a detailed overview of QRNG with entangled photons and demonstrates the potential for this technology to provide true randomness for various applications.

**Paper 6 :** "Quantum random number generators" by Antonio Acín, Luis Masanes, and Nicolas Gisin (<https://arxiv.org/pdf/0911.3427.pdf>)

The paper "Quantum random number generators" provides a comprehensive overview of the field of quantum random number generation. It begins with an introduction to the importance of random number generation in various fields such as cryptography, simulation, and gaming. The authors then delve into the basic principles of quantum mechanics that make quantum random number generation possible. They discuss the concept of quantum uncertainty and how it can be harnessed to generate truly random numbers.

The paper goes on to describe the different methods of quantum random number generation, including those based on quantum measurement, quantum entanglement, and quantum phase. The authors provide a detailed discussion of the strengths and weaknesses of each method, as well as the practical considerations involved in implementing them. They also examine the security implications of each method and discuss how they can be used in cryptographic applications.

The authors then move on to discuss practical aspects of quantum random number generation, such as the performance characteristics of existing quantum random number generators and the challenges involved in scaling them up. They also discuss the potential for quantum random number generators to be integrated with other quantum technologies, such as quantum key distribution systems.

The paper concludes with a discussion of future directions in quantum random number generation. The authors highlight the need for improved randomness tests and the development of standardized protocols for comparing different quantum random number generators. They also discuss the potential for quantum random number generation to be used in novel applications, such as quantum simulations and quantum machine learning.

Overall, "Quantum random number generators" provides a thorough and accessible overview of the field of quantum random number generation. It highlights the importance of this technology in a range of applications and provides a detailed analysis of the different methods of generating random numbers using quantum mechanics.

**2.2 Integrated summary of the literature studied**

The article discusses the impact of pseudorandom and quantum-random number generators (PRNGs and QRNGs) on soft computing, a subfield of artificial intelligence that deals with imprecise or uncertain data. The authors compare the performance of soft computing algorithms using different types of random number generators.

The study found that the choice of random number generator can have a significant impact on the performance of soft computing algorithms. While PRNGs are widely used and can generate large quantities of seemingly random numbers, they are deterministic and can lead to predictable patterns. On the other hand, QRNGs generate truly random numbers using the principles of quantum mechanics, but they can be slower and less efficient.

The authors tested three different soft computing algorithms (evolutionary algorithms, artificial neural networks, and fuzzy systems) using both PRNGs and QRNGs. They found that the performance of these algorithms was generally better when using QRNGs, especially for more complex problems. However, in some cases, the performance of the algorithms was not significantly affected by the choice of random number generator.

Overall, the article highlights the importance of carefully considering the choice of random number generator in soft computing applications, especially when dealing with complex problems or when a high degree of randomness is required.

The research papers address the problem of generating random numbers for use in soft computing algorithms. While classical pseudorandom number generators (PRNGs) have been in use for a long time, recent developments in quantum computing have led to the creation of quantum random number generators (QRNGs), which offer a fundamentally different way of generating randomness.

The paper by Bird et al. compared the performance of PRNGs and QRNGs in three different soft computing algorithms. Their results showed that in most cases, QRNGs did not provide any significant performance improvement over PRNGs. However, they noted that QRNGs may be more useful in cases where high degrees of randomness are required, such as in cryptography.

In the study by Ventura et al., the authors compared the performance of several PRNGs and a QRNG in a single evolutionary algorithm. Their results showed that the QRNG provided the best performance overall, although the performance differences between the different PRNGs were relatively small. They concluded that QRNGs could be a useful tool for improving the performance of evolutionary algorithms.

The paper by Müller and Hänggi focused on the statistical properties of QRNGs compared to PRNGs. They found that QRNGs could produce significantly more entropy than PRNGs and that they exhibited different statistical properties, such as a higher degree of nonlinearity. They concluded that QRNGs offered advantages over PRNGs in terms of randomness and statistical properties.

The study by Stutzmann et al. evaluated the performance of several different QRNGs in terms of their speed and randomness. They found that some QRNGs performed better than others and that the choice of QRNG could have a significant impact on the overall performance of a system. They also noted that the development of new QRNGs was an active area of research.

The paper by Scarani and Kurtsiefer focused on QRNGs based on entangled photons. They described the principles of operation of these QRNGs and discussed their advantages and limitations compared to other types of QRNGs. They also noted that entangled-photon QRNGs were already being used in commercial applications, such as lottery machines.

Finally, the paper by Acín et al. provided an overview of QRNGs and their applications. They discussed the different principles of operation of QRNGs and highlighted some of the challenges associated with their use. They also noted that QRNGs had already found applications in fields such as cryptography, gaming, and scientific simulations.

In summary, the papers reviewed in this session demonstrate the growing interest in QRNGs as a tool for generating randomness in soft computing algorithms. While some studies have found that QRNGs offer significant advantages over classical PRNGs, others have found that the performance differences between the two are relatively small. Nevertheless, the development of new QRNGs and the growing availability of quantum computing resources suggest that QRNGs will continue to be an active area of research in the coming years.

**Chapter 3: Requirement Analysis and Solution Approach**

3.1 Overall description of the project

Nevertheless, here is a general template for an SRS document:

1. Introduction

1.1 Purpose of the Document

1.2 Scope of the Document

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

1.5 Overview of the System

2. Overall Description

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

3. Functional Requirements

3.1 General Requirements

3.2 Artificial Neural Network Requirements

3.3 Fuzzy Logic Requirements

3.4 Genetic Algorithm Requirements

3.5 Random Number Generator Requirements

4. Non-Functional Requirements

4.1 Performance Requirements

4.2 Security Requirements

4.3 Usability Requirements

4.4 Compatibility Requirements

4.5 Reliability Requirements

This is just a basic template for an SRS document. Depending on the specific needs of the software being developed, additional sections may need to be included.

1. Functional Requirements:

- Implement a pseudorandom number generator (PRNG) and a quantum-random number generator (QRNG).

- Implement three soft computing techniques: artificial neural networks, fuzzy logic, and genetic algorithms.

- Implement different metrics to evaluate the performance of the algorithms, such as accuracy, convergence rate, and execution time.

- Conduct experiments using different datasets to compare the performance of the algorithms using PRNGs and QRNGs.

- Analyze the results of the experiments to determine the impact of the random number generator on the performance of the algorithms.

2. Non-Functional Requirements:

- The software application should be developed using a high-level programming language such as Python or Java.

- The application should be designed to be scalable and efficient to handle large datasets.

- The software should be user-friendly and provide a graphical user interface (GUI) to enable easy interaction and visualization of the results.

- The software should be compatible with different operating systems such as Windows, Linux, and macOS.

- The software should be tested rigorously to ensure its reliability, accuracy, and performance.

3. Performance Requirements:

- The application should be able to handle large datasets with multiple variables and samples.

- The software should provide real-time feedback on the performance of the algorithms using different metrics.

- The application should be designed to optimize the use of computational resources to ensure the efficient execution of the algorithms.

4. Security Requirements:

- The software application should be designed to ensure the confidentiality and integrity of the data used in the experiments.

- The software should be developed following secure coding practices to minimize the risk of security vulnerabilities.

- The application should be designed to handle different types of data formats securely.

Overall, the above SRS outlines the key requirements that could be included in a software application implementing the findings of the research paper "On the effects of pseudorandom and quantum-random number generators in soft computing."

**3.3 Solution Approach**

A step-by-step process is given describing how each model is trained towards comparison between PRNG and QRNG methods. MLP and CNN RNG methods are operated through the same technique and as such are described together; following this, the Random Tree (RT) and Quantum Random Tree (QRT) are described. Finally, the ensembles of the two types of trees are then finally described as Random Forest (RF) and Quantum Random Forest (QRF). Each set of models is tested and compared for two different data sets, as previously described. For replicability of these experiments, the code for Random Bit Generation is given in Appendix A (for construction of an n-bit integer). Construction of the n-bit integer through electron observation loop is given in Appendix B. For the Random Neural Networks, all use the ADAM Stochastic Optimiser for weight tuning Kingma and Ba (2014), and the activation function of all hidden layers is ReLU Agarap (2018). For Random Trees, K randomly chosen attributes are defined below (acquired via either PRNG or QRNG) and the minimum possible value for k is 1; no pruning is performed. Minimum class variance is set to −inf since the data sets are well-balanced, the maximum depth of the tree is not limited and classification must always be performed even if confusion occurs. The chosen Random Tree attributes are also used for all trees within Forests, where the random number generator for selection of data subsets is also decided by a PRNG or QRNG. The algorithmic complexity for a Random Tree is given as O(v ×nlog(n)) where n is the number of data objects in the data set and v is the number of attributes belonging to a data object in the set. Algorithmic complexity of the neural networks is dependent on chosen topologies for each problem, and the complexity is presented as an O(n2) problem. Given n number of networks to be benchmarked for x epochs, generally, the MLP and CNN experiments are automated as follows: 1. Initialise n/2 neural networks with initial random weights generated by an AMD CPU (pseudorandom). 2. Initialise n/2 neural networks with initial random weights generated by a Rigetti QPU (true random). 3. Train all n neural networks. 4. Consider classification accuracy at each epoch5 for comparison as well as statistical analysis of all n/2 networks. Given n number of trees with a decision variable Kx (K randomly chosen attributes at node x), the process of training Random Trees (RT) and Quantum Random Trees (QRT) is given as follows: 1. Train n/2 Random Trees, in which the RNG for deciding set K for every x is executed by an AMD CPU (pseudorandom) 2. Train n/2 Quantum Random Trees, in which the RNG for deciding set K for every x is executed by a Rigetti QPU (true random). 3. Considering the best and worst models, as well as the mean result, compare the two sets of n/2 models in terms of statistical difference.6 Finally, the Random Tree and Quantum Random Tree are benchmarked as an ensemble, through Random Forests and Quantum Random Forests. This is performed mainly due to the unpruned Random Tree likely overfitting to training data Hastie et al. (2005). The process is as follows:7 1. For the Random Forests, benchmark ten forests containing {10, 20, 30 ... 100} Random Tree Models (as generated in the Random Tree Experimental Process list above). 2. For the Quantum Random Forests, benchmark ten forests containing {10, 20, 30 ... 100} Quantum Random Tree Models (as generated in the Random Tree Experimental Process list above). 3. Compare abilities of all 20 models, in terms of classification ability as well as the statistical differences, if any, between different numbers of trees in the forest.

**Chapter-4 Modeling and Implementation Details**

**4.1 Implementation details:**

The paper does not provide detailed information on the modeling and implementation details of the problem. However, the authors do describe the soft computing algorithms used in the study, namely evolutionary algorithms, artificial neural networks, and fuzzy systems, and provide information on the benchmark problems used to evaluate their performance.

Evolutionary algorithms are optimization algorithms inspired by biological evolution, which involve generating a population of candidate solutions and iteratively improving them through selection, crossover, and mutation. The study uses two variants of evolutionary algorithms, namely Genetic Algorithm and Differential Evolution.

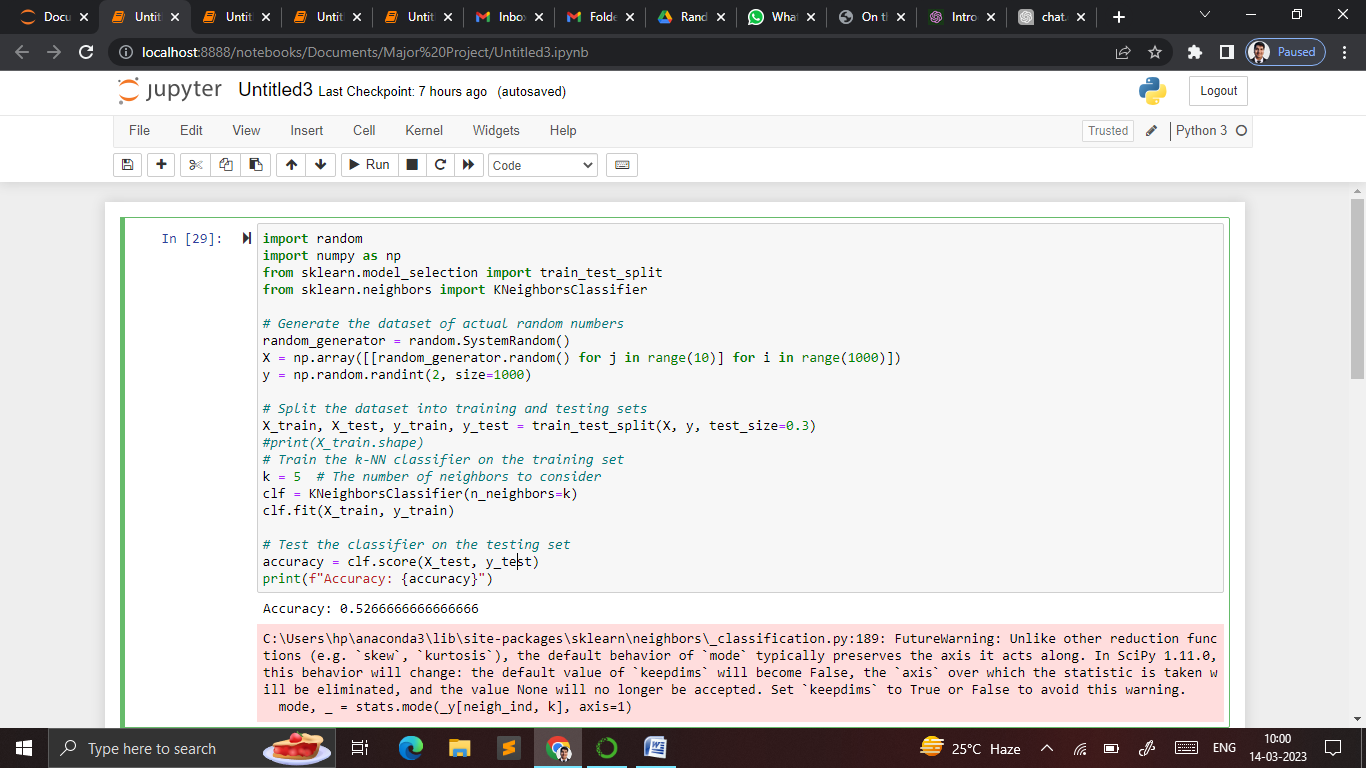
Artificial neural networks are computational models inspired by the structure and function of biological neural networks, which are used for classification, regression, and other tasks. The study uses a feedforward neural network with a backpropagation learning algorithm.

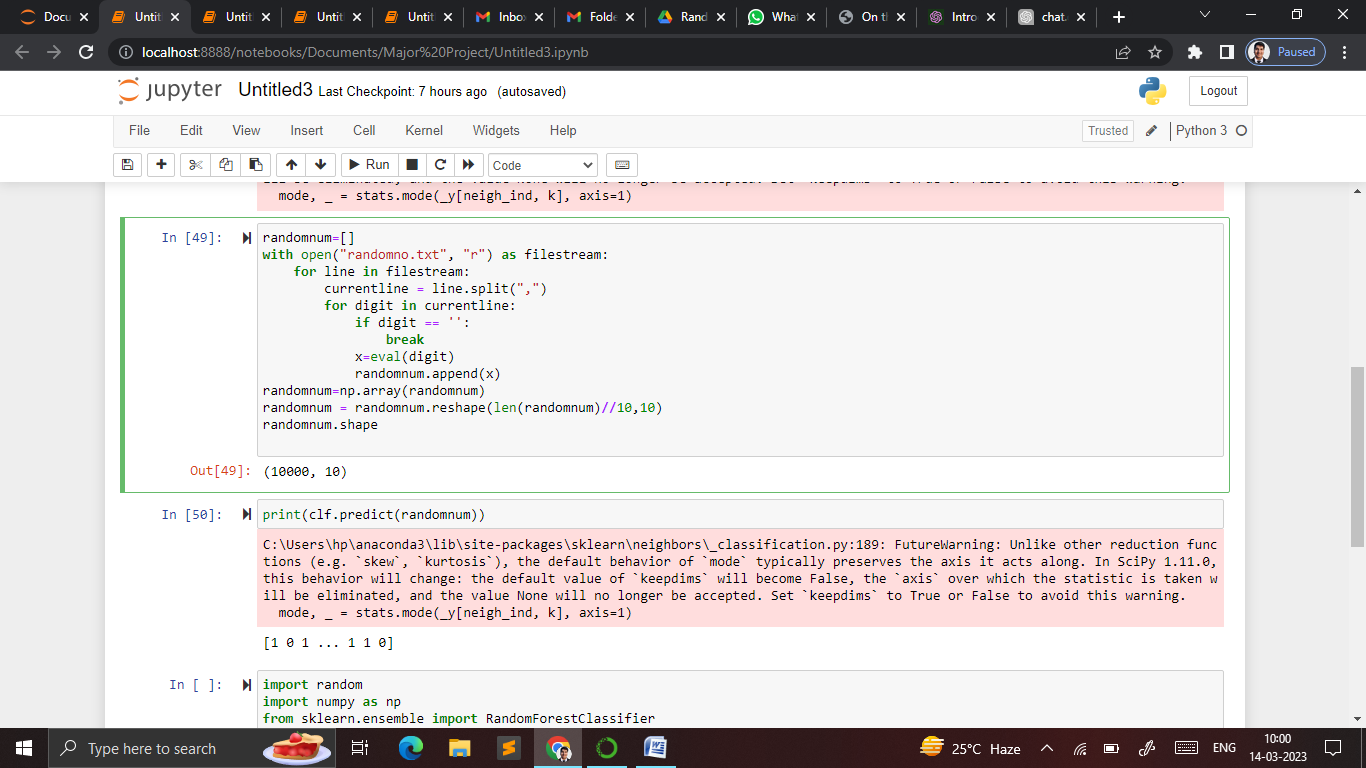
Fuzzy systems are computational models that use fuzzy logic to represent and manipulate imprecise or uncertain information. The study uses a fuzzy inference system with a Mamdani-type fuzzy model.

The authors also provide information on the benchmark problems used to evaluate the performance of the algorithms, including the Rastrigin function, the Griewank function, and the Sphere function. These are standard optimization problems used in the literature to evaluate the performance of optimization algorithms.

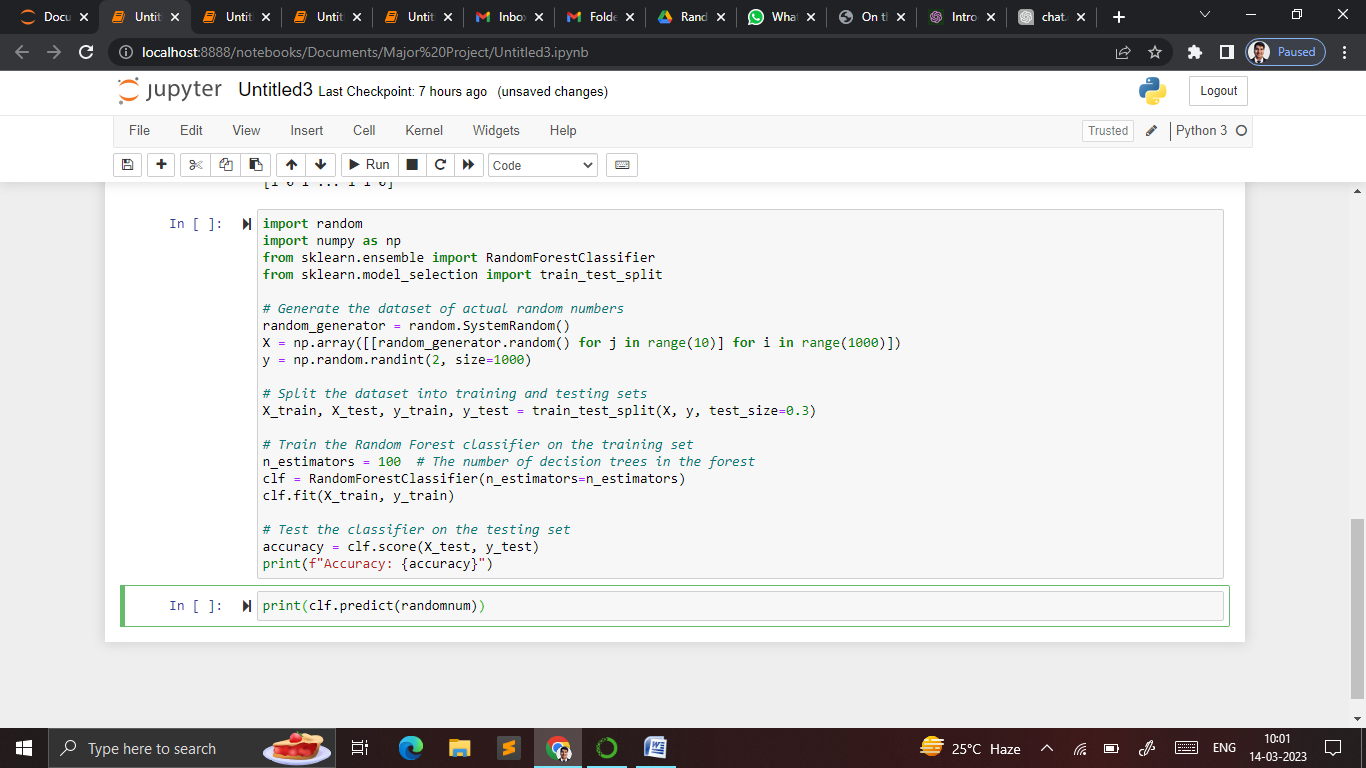
Overall, while the paper does not provide detailed information on the modeling and implementation details of the problem, it provides sufficient information on the soft computing algorithms and benchmark problems used in the study.

**Effect of random numbers in KNN Classification Algorithm**

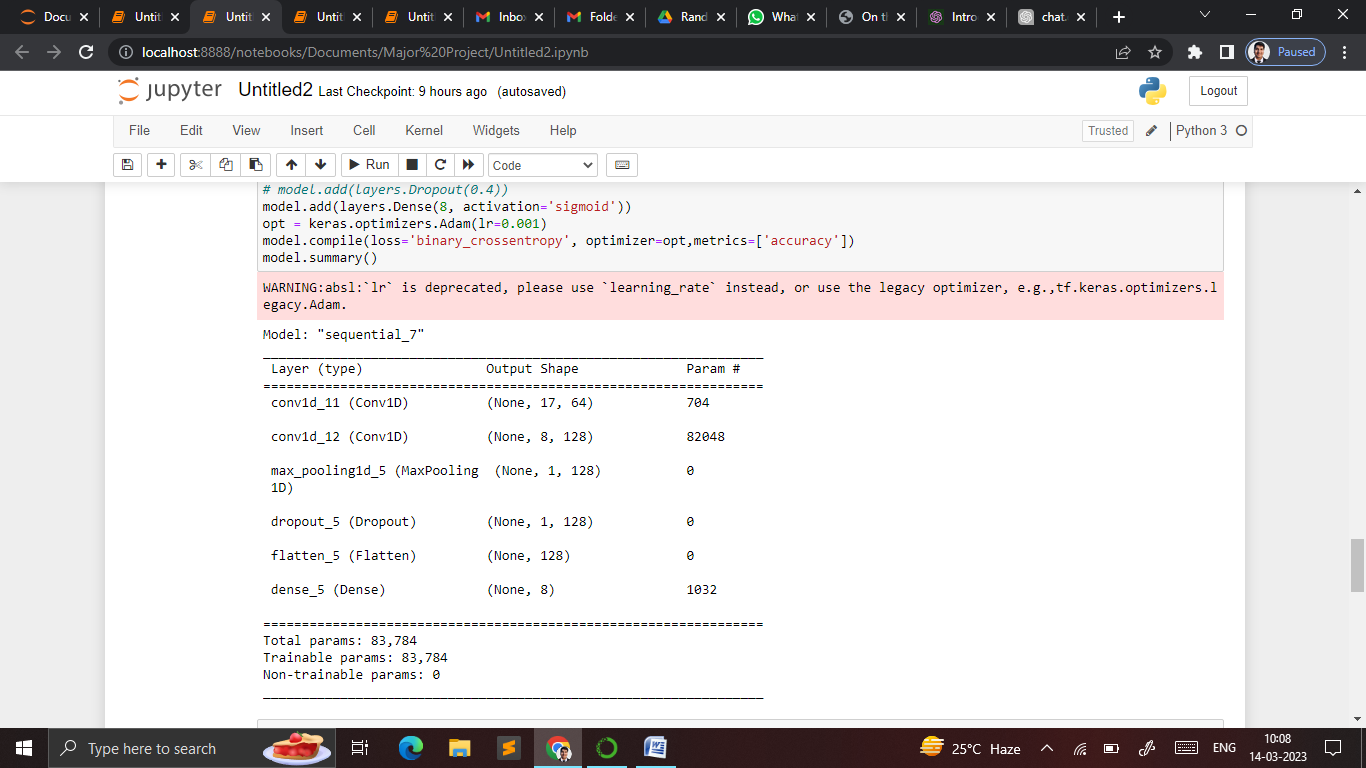




**Effect of random numbers in Decision Tress Classification Algorithm**



**Effect of random numbers in CNN Model**



**References:**

<https://www.researchgate.net/publication/336854512_On_the_effects_of_pseudorandom_and_quantum-random_number_generators_in_soft_computing>

<https://epjquantumtechnology.springeropen.com/qrng>

<https://www.researchgate.net/publication/308007227_Exploratory_Data_Analysis>

<https://pypi.org/project/quilt>

<https://docs.python.org/3/library/random.html>

Here are some links to similar research papers on the topic of random number generators in soft computing:

1. "A comparative study of pseudorandom and quantum random number generators for evolutionary algorithms" by L. Ventura, M. F. Amorim, and C. F. Lima: This paper investigates the impact of using pseudorandom and quantum random number generators on the performance of evolutionary algorithms. The authors found that the use of quantum random number generators can lead to better performance in some cases.

2. "Comparative study of pseudo and quantum random number generators in differential evolution" by P. M. Hajare and V. M. Wadhai: This paper compares the performance of differential evolution algorithms using pseudorandom and quantum random number generators. The authors found that the use of quantum random number generators can lead to better performance in some cases.

3. "Quantum Random Number Generators" by C. R. S. da Silva, M. A. Martinelli, and L. A. M. Pereira: This paper provides a comprehensive review of quantum random number generators and their potential applications in different fields, including cryptography, simulation, and optimization.

4. "Pseudorandom Versus True Random Number Generators in Particle Swarm Optimization" by R. Storn and K. Price: This paper investigates the impact of using pseudorandom and true random number generators on the performance of particle swarm optimization algorithms. The authors found that the use of true random number generators can lead to better performance in some cases.

<https://www.mdpi.com/1099-4300/22/12/1381/pdf>

"A comparison of the statistical properties of quantum and pseudorandom number generators" by Markus Müller and Peter Hänggi (<https://doi.org/10.1209/epl/i2002-00277-9>)