ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

THE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE BASED SEMI-AUTONOMOUS CONTROL SYSTEM TO ASSIST DECISION MAKING OF REACTOR OPERATORS

Ph.D. THESIS

Ceyhun YAVUZ

Department of Energy Science and Technology

Energy Science and Technology Programme

JULY 2025

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

THE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE BASED SEMI-AUTONOMOUS CONTROL SYSTEM TO ASSIST DECISION MAKING OF REACTOR OPERATORS

Ph.D. THESIS

Ceyhun YAVUZ
(301162001)

Department of Energy Science and Technology

Energy Science and Technology Programme

Thesis Advisor: Assoc. Prof. Senem ŞENTÜRK LÜLE

JULY 2025

ISTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

REAKTÖR OPERATÖRLERİNİN KARAR VERMESİNİ DESTEKLEMEK İÇİN YAPAY ZEKÂ TABANLI YARI-OTONOM KONTROL SİSTEMİNİN GELİŞTİRİLMESİ

DOKTORA TEZİ

Ceyhun YAVUZ
(301162001)

Enerji Bilim ve Teknoloji Anabilim Dalı

Enerji Bilim ve Teknoloji Programı

Tez Danışmanı: Doç. Dr. Senem ŞENTÜRK LÜLE

TEMMUZ 2025

Ceyhun YAVUZ, a Ph.D. student of ITU Graduate School student ID 301162001 successfully defended the thesis/dissertation entitled "THE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE BASED SEMI-AUTONOMOUS CONTROL SYSTEM TO ASSIST DECISION MAKING OF REACTOR OPERATORS", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Senem ŞENTÜRK LÜLE** İstanbul Technical University	.....
Jury Members :	**Prof. Dr. Gülgün KAYAKUTLU** Istanbul Technical University	.....
	Dr. Adem ERDOĞAN TENMAK	.....
	Prof. Dr. Üner ÇOLAK Istanbul Technical University	.....
	Assoc. Prof. Dr. Ali TİFTİKÇİ Sinop University	.....

Date of Submission : 15.06.2025
Date of Defense : 21.07.2025

To my family and ever beloved sister

FOREWORD

I would like to extend my heartfelt gratitude and appreciation to my thesis advisor Asst. Prof. Dr. Senem ŞENTÜRK LÜLE, who has supported and guided me in every step of the grueling processes and procedures with patience. It was a privilege and joy to work with and learn from her. This dissertation would not be possible without her efforts.

I would also like to thank Prof. Dr. Gülgün KAYAKUTLU who has guided and formed our efforts in implementing machine learning practices and Dr. Adem ERDOĞAN who has supported us with his vast knowledge in Relap simulations and nuclear energy implementation.

Prof. Dr. Atilla ÖZGENER and Prof. Dr. Bilge ÖZGENER who supported me during my Master's were essential on forming the basis of my nuclear engineering knowledge.

Finally, of course I would like to thank my parents who never stopped supporting me and believing in me after all these years and naturally my sister, Oytun YAVUZ, who is utterly dear to me.

Coşkun ÖZÜER, Çağlar KILIÇ, Deniz SEZER and Dr. Yetkin BORLU valuable friends all, you are also not forgotten, obviously.

June 2025

Ceyhun Yavuz
Physics Engineer

ABBREVIATIONS

AI : Artificial Intelligence
ANN : Artificial Neural Network
CNN : Convolutional Neural Network
DCNN : Deep Convolutional Neural Network
DRNN : Deep Rectifier Neural Network
FPR : False Positive Rate
GEP : Gene Expression Programming
GNV : Gaussian Naive Bayes
GPR : Gaussian Process Regression
KNN : K-Nearest Neighbor
LOCA : Loss of Coolant Accident
LOFA : Loss of Flow Accident
ML : Machine Learning
MLP : Multilayer Perceptron
NPP : Nuclear Power Plant
OOP : Object-Oriented Programming
PSO : Particle Swarm Optimization
PWR : Pressurized Water Reactor
R-CNN : Region Based Convolutional Neural Network
RCS : Reactor Coolant System
SVM : Support Vector Machine
TPR : True Positive Rate

SYMBOLS

f_s	**:** Feature Set
P(y)	**:** Relative Frequency of Class
a_i	**:** Lagrange Multiplier
β	: Equation Parameter
S	**:** Entropy

TABLE OF CONTENS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

THE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE BASED SEMI-AUTONOMOUS CONTROL SYSTEM TO ASSIST DECISION MAKING OF REACTOR OPERATORS

SUMMARY

The low-carbon energy sources are needed to combat climate change. Nuclear reactors will play a vital role in achieving climate change goals since electricity generation in nuclear power plants (NPPs) do not include greenhouse gas emissions. Although NPPs are safe to operate thanks to many safety systems both active and passive they contain, accidents are still possible with low probability. Determination of accidents is important not only for timely prevention but also for mitigation.

Four units of Russian Pressurized Water Reactor (PWR) namely VVER-1200 are under construction in Akkuyu site in Türkiye since 2018. The safety of these reactors are important for Türkiye. The accident detection study on VEER-1200 has paramount importance.

Machine learning (ML) is a subfield of artificial intelligence (AI) that enables computers to learn from data and improve their performance without needing specific programming for each task. Rather than relying on fixed rules, these systems detect patterns and trends in data to make predictions or decisions. Common applications include facial recognition, product recommendations, language translation, and fraud detection. ML methods include supervised learning (using labeled data), unsupervised learning (discovering patterns in unlabeled data), and reinforcement learning (learning through rewards and feedback).

This study focuses on identification of 53 transient sub-scenarios derived from reactivity insertion via rod withdrawal, steam leak from the pressurizer, loss of flow (LOFA), and loss of coolant accidents (LOCA) affecting both hot and cold legs main transients for VVER-1000 type PWR since the technical data of VVER-1200 is not readily available in the literature with ML. However, the methods developed in this study can be converted to VVER-1200 if data is provided.

Three approaches were introduced for the task. In one-step approach, data for all the sub-scenarios were used directly to identify transient sub-scenarios. In the two-steps approach, the main transients were identified first and then the sub-scenario identification was performed. In grouped one-step approach, with the guidance of confusion matrices of one-step approach, the sub-scenarios were grouped to increase the accuracy of the predictions.

The best performance of identification in one-step model was achieved with K-Nearest Neighbor (KNN) method which resulted in 74.66 % accuracy. This result shows that if all transient sub-scenarios were considered in one training method, there is a lot of confusion due to complex nature of nuclear system.

In two-steps approach, main transient detection with Random Forest method yielded the best result in terms of accuracy, precision, recall, and F1-score matrices, only slightly confusing LOCA hot leg transient with 1.44 %. This provided the sub-scenario identification almost 100 % accurate main transient base to optimize on. Sub-scenario identification with KNN method delivered 100% accuracy for rod withdrawal, steam leak from pressurizer, and loss of flow transients, providing the reactor operator an accurate understanding on the level of the transient. However, for LOCA transients, the method had a combined accuracy of approximately 80 %, which made the sub-scenario model usable for these transients, yielding a total of 86.44 % accuracy for two-steps approach.

In grouped one-step approach, new grouping with respect to transient levels resulted in 19 grouped sub-scenarios. This yielded a far better accuracy among all approaches with 92.04 % for KNN method. On the other hand, due to grouping, there is a loss of sensitivity in this approach.

For all approaches, the validation process was completed in 2-3 seconds, indicating that the approaches is sufficiently efficient and responsive for real-time monitoring applications.

REAKTÖR OPERATÖRLERİNİN KARAR VERMESİNİ DESTEKLEMEK İÇİN YAPAY ZEKÂ TABANLI YARI-OTONOM KONTROL SİSTEMİNİN GELİŞTİRİLMESİ

ÖZET

Düşük karbonlu enerji kaynakları, iklim değişikliğiyle mücadele etmek için gereklidir. Nükleer Güç Santrallerinde (NGS) elektrik üretimi sırasında sera gazı emisyonu olmadığından nükleer reaktörler iklim değişikliği hedeflerine ulaşmada önemli bir rol oynayacaktır. NGS'ler aktif ve pasif birçok güvenlik sistemine sahip olmaları sayesinde güvenle işletilselerde düşük olasılıkla da olsa kazalar meydana gelebilir. Kazaların belirlenmesi, sadece zamanında önlem almak için değil, aynı zamanda etkilerini azaltmak için de büyük önem taşır.

2018 yılından bu yana, Türkiye'nin Akkuyu sahasında dört adet Rus tipi Basınçlı Su Reaktörü (PWR) VVER-1200 ünitesi inşa edilmektedir. Bu reaktörlerin güvenliği Türkiye için büyük önem taşımaktadır. Bu nedenle VVER-1200 üzerinde yapılacak kaza tespiti çalışmaları son derece önemlidir.

Makine öğrenmesi, yapay zekâ alanının bir alt dalıdır ve bilgisayarların verilerden öğrenerek her görev için ayrı programlamaya ihtiyaç duymadan performanslarını geliştirmelerini sağlar. Sabit kurallara bağlı kalmak yerine, bu sistemler verilerdeki örüntüleri ve eğilimleri tespit ederek tahminlerde veya karar vermelerde bulunur. Yüz tanıma, ürün önerileri, dil çevirisi ve dolandırıcılık tespiti gibi uygulama alanları yaygındır. Makine öğrenmesi yöntemleri arasında denetimli öğrenme (etiketli veri kullanımı), denetimsiz öğrenme (etiketsiz verilerdeki desenlerin keşfi) ve pekiştirmeli öğrenme (ödül ve geri bildirim yoluyla öğrenme) yer alır.

Bu çalışma, VVER-1200 reaktörüne ait teknik verilerin literatürde yeterince bulunmaması nedeniyle, benzer bir reaktör tipi olan VVER-1000 için kontrol çubuğu çekilmesiyle gerçekleşen reaktivite eklenmesi, basınçlandırıcıdan buhar sızıntısı, akış kaybı ve sıcak/soğuk bacakları etkileyen soğutucu kaybı kazaları (LOCA) gibi senaryolardan türetilmiş 53 geçiş-durumu alt senaryonun makine öğrenmesi ile tespitine odaklanmıştır. Bu çalışmada geliştirilen yöntemler, gerekli verilerin sağlanması halinde VVER-1200'e de uygulanabilecektir.

Bu amaçla üç farklı yöntem önerilmiştir. Tek adımlı yöntemde, tüm alt senaryoların verileri doğrudan kullanılarak alt senaryo tespiti yapılmıştır. İki adımlı yöntemde, önce ana senaryolar belirlenmiş, ardından alt senaryo tanımlaması gerçekleştirilmiştir. Gruplanmış tek adımlı yöntemde ise, tek adımlı yöntemin karışıklık matrisinden elde edilen yönlendirme ile alt senaryolar gruplanarak tahmin doğruluğu artırılmıştır.

Makine öğrenme metdolarının eğitilmesi için kullanılacak 53 adet alt senaryoya ait veri seti, nükleer reaktörlerin simülasyonu için kullanılan ASYST programı kullanılarak elde edilmiştir. Reaktörlerin senaryo davranışları için ASYST programının kullandığı kod dosyasında hata kartlarını kullanmak sureti ile kazalar ve alt senaryoları simüle edilmiş, takip edilecek reaktör elemanı ve fiziksel parametlerine dair veri çıktısı sağlayan düzenleme kartları ile de 91 veri kolonu veya özellik simüle edilmiştir.

Çıktı dosyalarından veri setinin çekilebilmesi için Visual Basic dilinde iki seviyeli derleyici kod yazılmış ve bu sayede işlenebilecek şekilde excel programına yapılı bir fortmatta veriler aktarılmıştır.

Normalizasyon çalışmasını takiben, çalışmanın Python programında hayata geçirilebilmesi için, ilgili veri setleri .csv uzantılı dosyalara dönüştürülerek Google Colab derleme ortamına aktarılmıştır. Çalışma kapsamında Python Sci-Kit kütüphanesini kullanmak sureti ile K-En Yakın Komşu, Rastgele Orman, Gradyan Güçlendirme, Karar Ağaçları, Lojistik Regresyon, Destek Vektörü, Naïve Bayes makine öğrenme yöntemleri üzerinde modellerinin eğitim ve validasyon çalışması yürütülmüştür.

Tüm modeller için hiper parametre optimizayonu koşulmuş, veri setleri 80%-20% ve 70%-30% eğitim ve doğrulama setleri ile tutarlılık açısından test edilmiştir. Veri setin eğitim ve doğrulama setleri, tüm alt senaryolarda 80%-20% oranın tesis edecek şekilde, ancak aynı zamanda reaktörün SCRAM sürecinde farklılaşan güç gradyanı profillerinde de aynı oranda temsili sağlayarak hayata geçirilmiştir.

Tüm makine öğrenme modellerinin performansının değerlendirilmesi için doğruluk, hassasiyet, duyarlılık ve F1 Skorunu ihtiva edecek şekilde analizi yapılmış, makine modellerini hayata geçiren yöntemler ve modeller bazında da karışıklık matrisi değerlendirilmesi yapılmıştır.

Tek adımlı yöntemde en iyi başarı, %74,66 doğruluk oranıyla K-En Yakın Komşu (KNN) yöntemiyle elde edilmiştir. Bu sonuç, tüm alt senaryoların tek bir eğitim yöntemi ile değerlendirildiğinde, nükleer sistemlerin karmaşıklığı nedeniyle önemli ölçüde karışıklık yaşandığını göstermektedir. %74,66 doğruluk oranı, nükleer endüstrinin gerektirdiği yüksek performans gerekliliğini karşılamaktan uzak olduğu için, çalışma iki adımlı yönteme doğru yönlenmiştir

İki adımlı yöntemde, ana geçiş-durumu senaryosu tespitinde Rastgele Orman yöntemi, doğruluk, kesinlik, duyarlılık ve F1 puanı açısından en iyi sonucu vermiştir; yalnızca sıcak bacak LOCA senaryosunu %1,44 oranında karıştırmıştır. Bu sayede, alt senaryo tespiti neredeyse %100 doğru bir ana senaryo temelinde gerçekleştirilmiştir.

Ana geçiş-durumu senaryosuna dair 99.51% doğruluk performansı gösteren Rastgele Orman yöntem, iki adımlı yöntemin birinci kısmını oluşturmuştur. Takiple tespit edilen ana geçiş-durumunun alt senaryosunu tespit etmek için her ana geçiş için bir alt senaryo tespit modeli geliştirilmiştir.

Alt senaryo tespitinde KNN yöntemi, çubuk çekilmesi, basınçlandırıcıdan buhar sızıntısı ve akış kaybı senaryoları için %100 doğruluk sağlamış ve operatöre geçişin seviyesi hakkında net bilgi sunmuştur. Ancak LOCA senaryolarında yöntem yaklaşık %80 doğruluk sağlamış ve genel olarak iki adımlı yöntemde %86,44 doğruluk elde edilmiştir. Bu anlamda nükleer sektörün yüksek standardına daha yakın perfomans ortaya konmuştuır.

Yapılan çalışmalarda karışıklık matrisi tüm ana geçiş-durumları ve alt senaryolar için incelenmiş ve karışıklığın çok sıklıkla, özellikle LOCA senaryolarında, senaryo şiddeti anlamında 1 seviye yukarıda veya aşağıda olduğu görülmüştür. Buradan hareket ile, bu karışıklık skorlarına kapsayacak şekilde yapacak daha geniş bir gruplamanın, hassasiyet kaybına sebep olacak ise de, daha yüksek bir doğrulukla alt senaryo tespitine imkan vereceği değerlendirilmiştir.

Gruplanmış tek adımlı yöntemde, geçiş-durumu seviyelerine göre yapılan yeni gruplamayla 19 alt senaryo kümesi oluşturulmuştur. Bu, KNN yöntemiyle %92,04 doğrulukla tüm yöntemler arasında en yüksek başarıyı sağlamıştır. Ancak, gruplama nedeniyle hassasiyette bir miktar kayıp yaşanmıştır.

Tüm yaklaşımlar için doğrulama süreci 2–3 saniyede tamamlanmış ve bu da yöntemlerin gerçek zamanlı izleme uygulamaları için yeterince hızlı ve etkili olduğunu göstermektedir.

Yapılan çalışma kapsamında gelecek dönemde makine öğrenmesine ek olarak, yapay zeka ve derin öğrenme modellerinin de, alternatif olarak ele alınması gerektiği ve makine öğrenme modellerine karşı performanslarını değerlendirlmesinin gerektiği tartışılmıştır.

Yine gelecek çalışmalarda 91 adet veri kolonunun, simülatör çalışması kapsamında elde edilmesi kolay olsa da, gerçek operasyonda bu sayıda ve detayda veriye tam zamanlı ve sürekli ulaşmanın sorun olacağından hareket ile, aynı veya daha fazla doğruluk skoruna, daha az veri kolonu ile erişmenin gerekliliği değerlendirilmiştir.

Daha verimli ve değerli bir model önermesinde de, ASYST kodun nodalizasyonun izin verdiği ölçüde reaktörde alt senaryoların lokasyon bilgisi ile kartezyenleştirilerek çoğaltılması ve makine öğrenmesi veya yapay zeka modeli yardımı ile hem alt senaryonun hem de lokasyonun aynı anda tespit edilmesi ele alınmıştır. Böylece modelin sadece reaktör operatörlerine değil aynı zamanda reaktörün sorun gidermekten sorumlu kadrosunun da verimliliğine katkı verebileceği düşünülmektedir.

1. INTRODUCTION

1.1 Purpose of Thesis

Nuclear reactors play a fundamental role as a global energy supply system, yet they also carry significant risks with severe consequences in the event of accidents. Therefore, the imperative of ensuring their safe operation is paramount. During transient conditions, the values of key parameters may fluctuate in ways that they are not immediately visible. Therefore, the detection of these transients is essential for timely preventive measures and subsequent protective actions.

1.2 Literature Review

AI has shown promise in many fields, and its potential in the nuclear field has gained increasing interest in recent years. Various applications have drawn interest including problems related to nuclear reactor modeling, fuel cycle, fault diagnosis, and treatment of radioactive waste [1]. Substantial efforts have been devoted to lumped-parameter modeling, fluid dynamics, system code development, and application of ANNs in nuclear reactor thermal-hydraulics and diverse specialties such as neutron transport, thermal hydraulics, and fuel performance. ML is an application alternative to tackle previously intractable problems in science and engineering, concentrated in defect detection, dose prediction, component failure detection, core and transient monitoring, gamma spectroscopy, and in-core fuel optimization fields.

1.2.1 Defect detection in nuclear fuel assembly

Fuel assemblies are essential components of nuclear reactors, fulfilling both structural and functional roles. They vary in design, but most commercial power plants utilize fuel assemblies consisting of fuel pellets securely contained within fuel rods. Under standard operating conditions, these assemblies are engineered for stability and are periodically inspected for leaks, swelling, or bowing as any complications could significantly affect environmental safety [2]. Potential defects might be hidden in simple issues like deficient welds or in serious issues like corrosion. Cladding defects

are particularly important, as they are subjected to radiation and corrosive coolants. Cladding cracks may arise from multiple factors, including material overstress or inadequate cooling practices [3]. Fuel assemblies consist of components such as a bottom nozzle and spacer grids. The nozzles and spacer grids are significant elements with complex welds, presenting challenges for manual inspection, and defects in these components can pose serious safety risks. Likewise, the fuel rods can experience stress from temperature variations and radiation, leading to structural deformation. Corrosion is another complex category of damage, wherein both internal and external corrosion can result in material thinning due to numerous possible causes. Service inspections are carried out using manual and visual techniques, often aided by cameras. This method is labor-intensive and open to subjective interpretation, hence harbors possibility of erroneous procedure. An alternative approach is the sipping test, which detects defects in a fuel assembly by measuring the concentration of the Kr-85 in a sample; however, this cannot be utilized on assemblies that are operational. For better performance of defect detection, a deep convolutional neural network (DCNN) was trained to identify scratches on nuclear fuel assemblies. This framework achieves a True Positive Rate (TPR) of 0.98 while maintaining a False Positive Rate (FPR) of 0.1, outperforming many existing detection methods. [4]. A proof of concept for the region based convolutional neural network (R-CNN) shown in Figure 1.1 displays the model's reliability, illustrating its ability to accurately distinguish between actual defects—resulting in a high TPR—and water stains or other false positives, thus contributing to a low FPR [4]. Optimization of better image recognition for defect detection has also been utilized for KNN, ANN [5]., SVM and GNV [6]. in recent literature, diversifying the methods for research.

Figure 1.1: The detected image of testing data [4].

1.2.2 Dose prediction in nuclear emergencies

Dose prediction during nuclear emergencies is crucial in the event of accidental radioactive material release into the environment. The effects of ionizing radiation exposure are linked to the absorbed dose in human tissues, which is influenced by factors such as dose rate, duration of exposure, and individual protection capabilities. It is essential to provide updated dose assessments to the population, whether general or specific, for protective measures and to ensure understanding of critical guidelines, particularly concerning early protective actions near affected areas. Quick and dynamic operations are necessary to analyze relevant meteorological and site data. Additionally, factors related to population behaviors and movements are significant. A notable challenge in addressing the physical aspects related to this issue is the reliance on low-technology methods due to inherent flaws in proposed methodologies. The anticipated result is a better understanding of dose distribution, as well as improvements in certainty, accuracy, and speed related to collective exposure scenarios [7]. AI and ML can transform dose prediction practices in the aftermath of nuclear or radiological emergencies. AI and ML can effectively generate predictive models from data, enabling continuous learning and adaptation [8]. During and in the

aftermath of a nuclear or radiological crisis, ML can be beneficial, especially if the data that is utilized for analysis is substantial in size. These algorithms facilitate the processing of extensive datasets, identifying statistically significant patterns that enhance decision-making. Simulated dosimetric data often serve as another important data source. AI and ML can develop predictive models based on this simulated data, which can subsequently be applied to real event scenarios. The Deep Rectifier Neural Network (DRNN) was evaluated to predict spatial effective doses under different atmospheric conditions, utilizing two realistic accident scenarios modeled with the atmospheric dispersion system from a Brazilian NPP. In the simpler scenario, the optimal DRNN outperformed the 5 Multi-Layer Perceptron (5- MLP) model, achieving a 25% reduction in error and completing training 155 times faster. In the more complex scenario, the top DRNN reached an average error of 0.0213 with a training duration of 30 minutes, demonstrating the capability of DRNNs to improve dose prediction based on ANNs in real-world applications [9]. Work for prediction of dose through other ML methods like PSO, GPR [10]. and CNN [11]. has been applied progressing the ML and AI methods further.

1.2.3 Fuel and component failure detection

In an NPP, fuel cladding serves the purpose of containing radioactive materials within a controlled system. If this component fails, the discharge of radioactive materials into the primary loop is unavoidable. A prevalent technique for detecting such failures is the isotopic ratios method, which involves monitoring the proportion of released isotopes present in the coolant [12]. An increase in this ratio above a defined threshold indicates that a failure has occurred. A four-layer ANN has been proposed for fuel failure detection, featuring an input layer, two hidden layers, and an output layer, as illustrated in Figure 1.2. The neurons process the total values received from the previous layer and pass them to the subsequent layer using an activation function. The inputs to the ANN consist of normalized specific activities of 23 common fission products, including Br-83, Kr-85, Kr-85m, Kr-87, Kr-88, Sr-90, Te-131, Te-131m, I-129, I-131, I-134, I-135, Xe-133, Xe-133m, Xe-135, Xe-135m, Xe-138, Cs-134, Cs-134m, Cs-137, and Cs-139 [13]. ANN proves to be more responsive when the fuel cladding is defective.

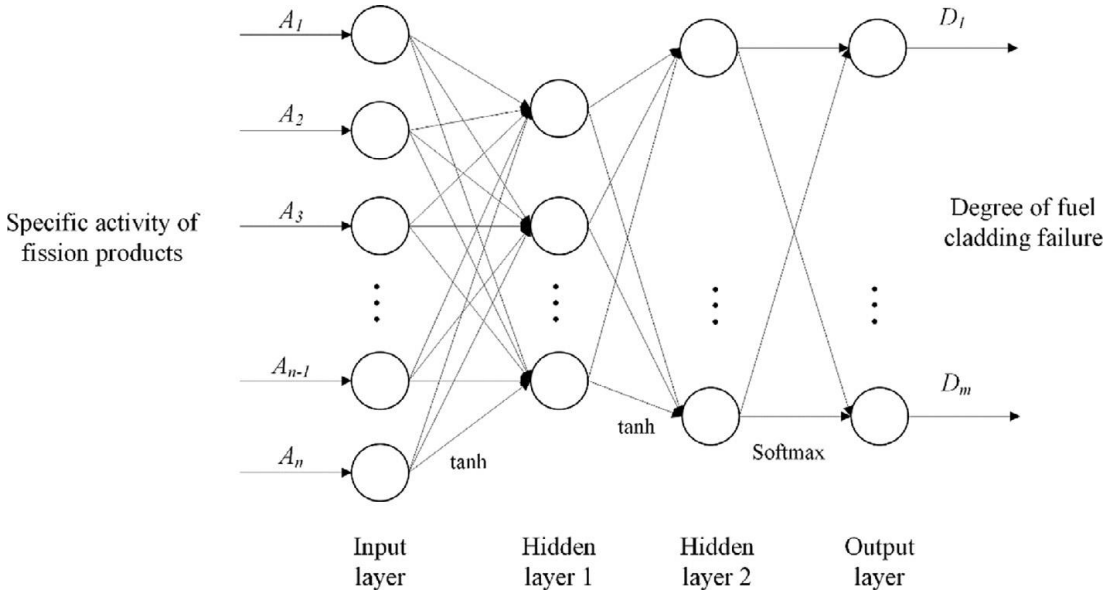

Figure 1.2: ANN Schematic for fuel failure detection [13].

1.2.4 Core monitoring for reactor and transients

A real-time monitoring system for the power distribution within a 3D reactor is essential for ensuring nuclear safety and enhancing the operational efficiency of NPPs. This system is particularly important for optimizing control mechanisms, particularly when the NPP is operating at a specific power output or engaged in load-following operations [14]. Additionally, NPPs may undergo transient events caused by equipment failures or dysfunctions in processing systems. In such situations, operators are required to implement diagnostic and corrective measures in response to the identified transients. ANNs facilitate rapid diagnostics while maintaining an acceptable level of error. This capability minimizes the risk of human evaluation errors and allows for swift responses to existing scenarios. A benchmark measuring the accuracy of transient detection and transient is provided in Table 1.1. By reducing error rates to an acceptable threshold, the implementation of a decision support system becomes both viable and actionable [15]. Several AI studies were performed for transient detection not only for the LOFA and LOCA accidents which are the majority of the literature but also for steam generator tube rapture and power change, for VVER 1000 reactors specifically; Random Forest was utilized to detect LOCA accident and its exact size, location in the reactor with an accuracy of 97.26% [16]., GEP method has been applied for LOFA detection with an accuracy of 99% [17]., 15 features has been trained for detection of 10 transients with different ANN algorithms (Resilient

backpropagation, Standard backpropagation etc) scoring an accuracy of 96.75 % for all transients combined [18]., and local blockage accidents were detected with 90.1 % accuracy employing neural network [19].

Table 1.1: Transients and detection error [10].

Transient	Detection Error (%)
SG1TR (Steam Generator)	4.5
SG2TR (Steam Generator)	5.2
SG3TR (Steam Generator)	4.9
SG4TR (Steam Generator)	4.6
MODHX1TR (Moderator heat exchanger)	5.3
MODHX2TR (Moderator heat exchanger)	5.5
Bleed cooler tube rupture	7.4
Shut down cooling heat exchanger tube fail	8.2

1.2.5 Gamma spectroscopy analysis

Neutron activation analysis (NAA) is a technique utilized by quantitative and qualitative analysis of elements in various samples, relying on the detection of distinct radiation pattern, emitted by radionuclides produced when materials are irradiated with neutrons. The ANN algorithm can effectively analyze gamma-ray spectra by identifying patterns associated with the energies of emitted gamma rays [20]. If only one gamma-ray is emitted from a radioisotope and its energy coincides with that of other gamma rays, however, the ANN may encounter challenges in accurately identifying the specific radioisotope involved [21].

1.2.6 In-core fuel management models

Due to limitations related to fuel burn-up, the processes of fuel loading and shuffling typically occur on an annual basis though it might be longer than a year. During the refueling process, fuel assemblies that are used for a long time in the reactor, are removed from the reactor core, and remaining assemblies are rearranged while new fuel assemblies are introduced. The primary objective of refueling is to enhance the effective multiplication factor (k_{eff}) while ensuring that the power peaking factor remains within operational limits [22]. Over time, various algorithms have been developed to identify fuel configurations (in terms of positions and fuel designs) that optimize the coefficients of a defined objective function. Although these problems can differ significantly in terms of specifics and functional forms, they exhibit common

requirements for the search algorithm. The search space is extensive, encompassing 220-400 positions in the core. The objective surfaces tend to be non-linear, characterized by narrow optima and multiple extrema. Additionally, the objectives are subject to a range of physical and design constraints, which may pose substantial challenges in meeting them. While designers may possess an intuitive understanding of "good" solutions, the complexity of the solution space complicates the identification of optimal solutions through conventional methods such as "hill-climbing," "brute force," or "trial and error" techniques. Particle Swarm Optimization (PSO) has been employed to develop a model that optimizes both peaking factor and k_{eff} according to the achieved configuration. As seen in Figure 1.3 This approach resulted in an increase in k_{eff} by 8.79% and a decrease in power peaking by 0.047% relative to the standard reference loading pattern [23].

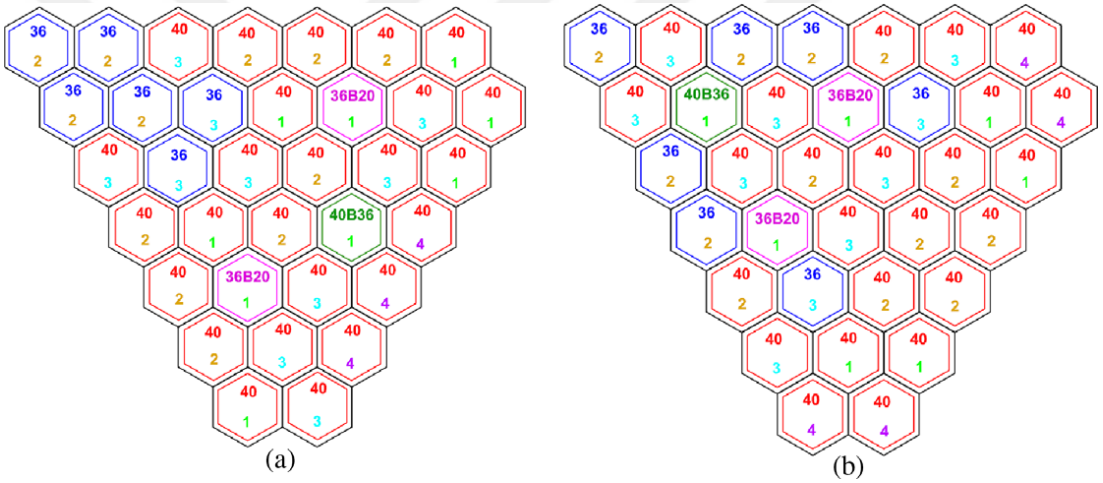

Figure 1.3: (a) Particle swarm proposition for core loading pattern, (b) Standard core loading pattern [23].

2. VVER 1000 AND VVER 1200 REACTORS

VVER, a family of PWRs, were first initiated by OKB Gidropress, which is a satellite of the ROSATOM, a nuclear firm of Russian Federation. The development of nuclear plants utilizing VVER infrastructure has been undertaken by institutions within ROSATOM, including Moscow ATOMENERGOPROEKT (AEP), St. Petersburg AEP, and Nizhny Novgorod AEP [24]. Construction of the first VVER NPP began in the 1960s, culminating in the commissioning of the Novovoronezh NPP in 1964, which housed the V-210 unit [25]. The subsequent unit at Novovoronezh-2 was designated V-365, with these identifiers originally correlating to the respective electrical outputs of the main generators. Since that time, VVER reactors have made substantial progress, with approximately 67 units constructed in various countries, including Armenia, Bulgaria, China, the Czech Republic, Finland, Hungary, India, Iran, Slovakia, Ukraine, and predominantly Russia, their country of origin. The successful deployment of two units at Novovoronezh established a foundational basis for the development of more advanced reactors. This led to the development of VVER Generation-II, introducing features like emergency core cooling, auxiliary feed water systems, and improved accident localization systems. VVER-440 reactors have been operated without breaching necessities of safety across several countries. Utilizing the know-how by Generation-II reactor operations, Generation-III reactor development and design efforts took place, resulting in the VVER-1000, which has emerged as the primary design in the VVER series.

2.1 VVER-1000

The inaugural VVER-1000 reactor was built at Novovoronezh 5 (VVER-1000/V-187) and commenced operations in 1981 [26]. This unit is equipped with a containment structure designed to endure maximum pressure during a design basis accident, including a failure of the main circulation pipeline with a standard diameter of 850 mm. The design concepts used in VVER-1000/V-187 were later applied to smaller series reactors, such as VVER-1000/V-302 (South Ukraine 1) and VVER-1000/V-338

(Kalinin 1 and 2), and subsequently to the larger series VVER-1000/V-320, which encompasses facilities like Zaporozhye, Balakovo, Kalinin 3 and 4, Rostov, Kozloduy, and Temelin, among others [27].

As illustrated in Figure 2.1, it includes a pressure vessel itself, four circulation loops, main coolant pumps, steam generators, a pressurizer, and a make-up and emergency core cooling systems. The primary coolant is responsible for extracting the heat produced by fission in the reactor core. All circulation loops are the same, while the pressurizer is linked to loop number 2, which manages reactor pressure level. The main coolant pumps of the reactor ensure an adequate flow rate of the primary cooling fluid. Subsequently, the coolant is circulated through the primary side of the steam generator and facilitates heat transfer to the secondary side. The overall characteristics of the VVER-1000 plant utilized in the modeling are presented in Table 2.1.

Table 2.1: VVER 1000 design parameters [28].

No	Main Category	Sub Category	China Tianwan NPP Units 3&4
1	**Design**		VVER-1000/V-428
2	**General Information**	General Designer	ATOMPROEKT, St.-Pt
		Nuclear Island Supplier	OKB Gidropress
		Turbine Island Supplier	Harbin Electric Company Ltd
		Construction	since 2012
		Commerical Operation	since 2018
3	**Reactor Power**	Thermal (MWt)	3012
		Electrical (MWt)	1126
4	**Fuel**	Max Enrichment	4.10%
		Fuel Assembly Type	TVS-2M
		Fuel Asseblies Number	163
		Control Rods Number	121
		Max Burn-Up	45.9 MW day/kgU].
5	**Spent Fuel Pool**	Capacity Total	706 fuel assemblies
		Water Inventory	~ 1700 m^3
		Heat Exchanger Power of Cooling System	~17 MW
		Measures for SBO Accidents	RHR system of spent fuel pool
		Monitoring Instrumentation for BDBA Conditions	Water Level Detection for Spent Fuel Pool

Table 2.1 : Table 2.1: VVER 1000 design parameters [28]. (Continue)

No	Main Category	Sub Category	China Tianwan NPP Units 3&4
6	Reactor Recirculation System	Number of Loops	4
		Main Coolant Pump	22000 m^3/h,
		Nominal Pressure in Primary Circuit	15,7 MPa
		Design Pressure in Primary Circuit	17,64 MPa
		Reactor Outlet Temperature	321 C
		LBB Concept Measures Implementation	For main coolant and ECCS passive part pipelines
7	Control rods system		CPS CR drive (SHEM-3)
8	Steam generator	Number and Type	4×PGV-1000M
		Steam Capacity	1470 t/h
		Design Lifetime	40 years
		Safety Valves Relief	2 per steam generator
		Nominal Steam Pressure	6,27 Mpa
		Design Pressure	7,84 Mpa
		Outlet Steam Temperature	278,5 C

Table 2.1: VVER 1000 design parameters [28]. (Continue)

No	Main Category	Sub Category	China Tianwan NPP Units 3&4
9	**Safety Systems**	Emergency Boron Injection System	4 x 50%
		Emergency Steam-Gas Removal System	4 x 100%
		Emergency Core Cooling System of High-Pressure (HP ECCS)	1 pump/channel
		Emergency Core Cooling System Of Low Pressure (LP ECCS)	1 pump/channel
		Hydro-Accumulators	4x 50 m^3 of each
		Automatic Pressure Relief System	3 PRZ PORV
		Emergency Feed Water System or Emergency Cool Down System	1 water tank and pump per each channel
		Steam Generator Heat Removal System	None
		Emergency Heat Removal Means	Pumps 80 m3/h to spent fuel pool and steam generators
		Hydrogen Recombiners	44
		Spray System in Containment	4 x 50%
10	**Means For Residual Heat Removal**		JNA (RHR system) pipelines containment spray pumps

Table 2.1: VVER 1000 design parameters [28]. (Continue)

No	Main Category	Sub Category	China Tianwan NPP Units 3&4
11	**Containment**	Type of Containment	Double containment
		Height	64.5 m
		Inner diameter	44.0 m
		Design pressure	0.5 MPa
		Leak tightness	<0.3% at design pressure
12	**Corium Catcher**		Steel+multicomponent system
13	**Turbine**	Type	TC6F-54
		Thermal cycle	HPC + 3LPC
		BRU-K (turbine bypass valve) Output	BRU-K 60 %
14	**Electrical equipments**	Direct Current Power Supply	EPS 4 x 400 Ah CPS 1 x 2600 Ah
		Generator with the Internal Combustion Turbine Drive	150 kW mobile DG
		Diesel Generator	4 x 5500kW

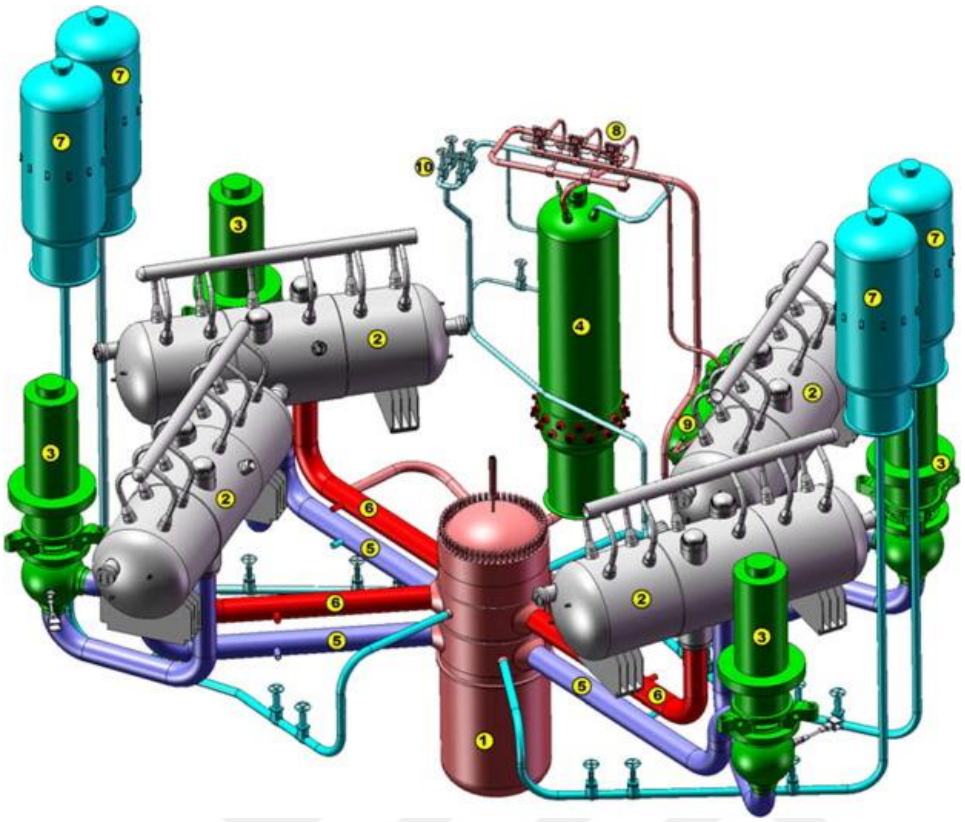

Figure 2.1: Primary circuit of VVER-1000 plant; (1) pressure vessel, (2) SG, (3) main coolant pump, (4) pressurizer, (5) cold leg, (6) hot leg, (7) accumulator, (8) PRZ pulse safety device valve, (9) relief tank, (10) injection system [28].

2.2 VVER 1200

The VVER-1200, also known as NPP-2006 or AES-2006, represents an advancement of the VVER-1000, intended for both domestic and international markets. This reactor design has undergone enhancements to improve fuel efficiency. The reactor achieves a net thermal efficiency of 34.5 %. The VVER-1200 is capable of generating 1,198 MW of electrical power. It is designed for a lifespan of 60 years, with options for a 20-year extension. The initial two units have been constructed at the Leningrad Nuclear Power Plant II and Novovoronezh Nuclear Power Plant II. Additional reactors based on the VVER-1200/491 design, similar to the Leningrad-II model, are also in the planning and construction phases, including projects in Kaliningrad and Nizhny Novgorod [29]. VVER-1200 type reactors have been built in Belarus, Russia, and Finland, and are planned to be built in other countries. Four units are under construction in Türkiye Akkuyu site near Mersin. The VVER-1200 nuclear reactor integrates the design features of the previous generation models and new solutions that ensure the achievement of the stated goals and reduce the risks of operation defects.

The advantages of the well-developed and technically sound automated manufacturing technologies of the VVER-1000 reactor vessel are successfully used [30].

Design parameters for VVER-1200 St-Petersburg power plant has been presented in Table 2.2 and components in Figure 2.2.

Table 2.2: VVER 1200 Design Parameters [31].

VVER-1200 (AES-2006) – the basic data, for the St Petersburg AEP	**Parameters**
Service life	60 years
Electricity output	1198 MWe gross
Reactor thermal output	3212 MWt
Heat supply capacity	300 MWt
Availability	>90%
Power plant efficiency	34.5%
Unplanned automatic scram per year	<1/year
Planned outage duration (annual) over seven years of operation	416/224/130 days, max
Duration of outage required every eight years to include turbine disassembly	40 days, max
Number of operating personnel	0.42 person/MW
Design basis maximum fuel burn-up	60 MWd/kgU per fuel assembly
Fuel campaign duration	4 Years
Refuelling frequency	12(18) Months
Primary coolant temp. at core inlet	298.2 °C
Primary coolant temp. at core outlet	328.9 °C
Primary coolant flow rate through reactor vessel	86000 m^3/hour
Primary coolant pressure at reactor vessel outlet	16.2 MPa
Steam pressure at the steam generator outlet	7 MPa
Steam production rate per SG	1602 t/hour
Feed water temperature at SG inlet	225 °C
Steam moisture content at SG outlet	<0.2%
Total probability of core damage due to internal initiating events	$<7.37\times10^{-7}$/ reactor year
Total probability of accidental sequences involving large releases caused by containment bypass or initial lack of leak tightness	$<3.71\times10^{-9}$
Double containment dimension	
External, protective, containment (reinforced concrete):	
Internal diameter	50 m
Height of dome	71.4 m
Thickness (cylindrical section)	2.2 m
Thickness (dome part)	0.8 m
Internal, hermetic, containment (also reinforced concrete):	
Internal diameter	44 m
Height of the dome	67.1 m
Thickness (cylindrical section)	1.2 m
Thickness (dome part)	1.1 m

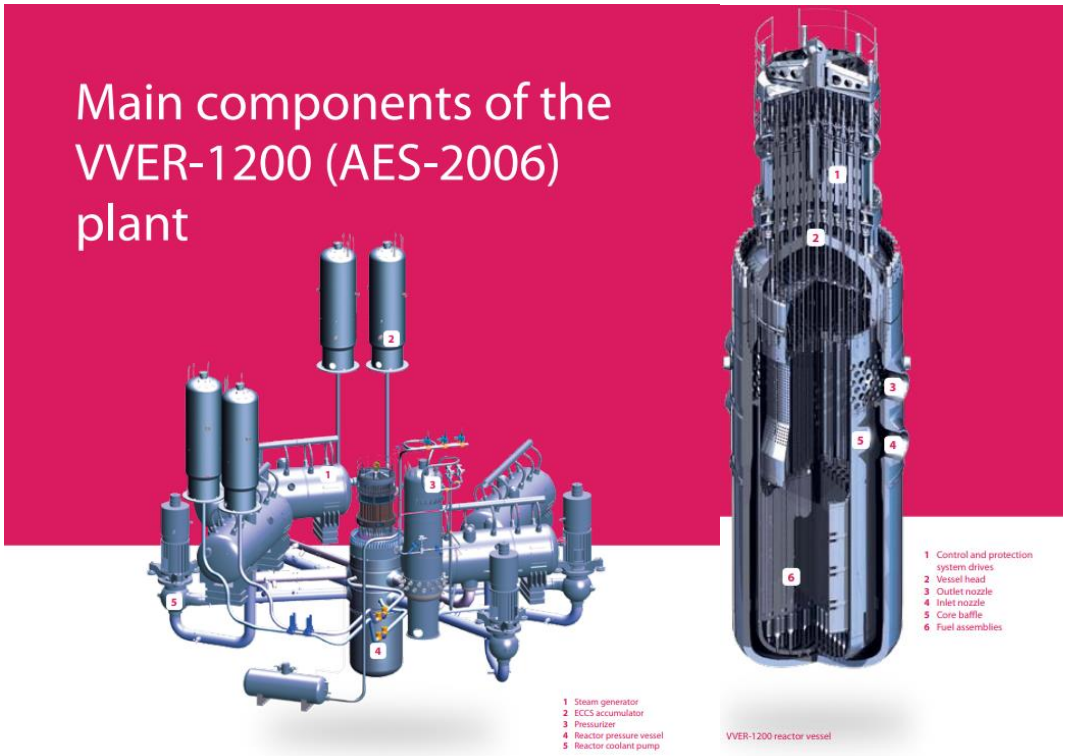

Figure 2.2: Main Components of VVER-1200 plant and Reactor Vessel [31].

3. MACHINE LEARNING

Rule-Based Expert systems, being valid approaches to control a physically isolated or informatics system, falls short of distinguishing between co-dependent physical parameters and deducing an actual decision. ML is a subdivision of AI that aims to utilize computers as a medium to simulate human learning, which makes it possible for them to execute processes independently while increasing precision by continuously learning through a large data set [32, 33]. Methods for ML used in this study were selected with respect to the intersection of commonly applied methods in nuclear reactor transient detection in literature and availability of methods in Sci-Kit Library.

3.1 K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a widely utilized classification technique in ML that operates under a supervised learning framework. It determines the class of a test data point by evaluating the majority vote among its nearest neighbors. The classification process is grounded in the feature space axes, where data points are categorized based on their proximity to their nearest neighbors. The distances between these points are measured using specific distance metrics pertinent to the feature space. To assess the similarities among data points, various distance metrics are employed to compute the distances, with Euclidean, Manhattan, and Minkowski being among the most commonly used [33, 34].

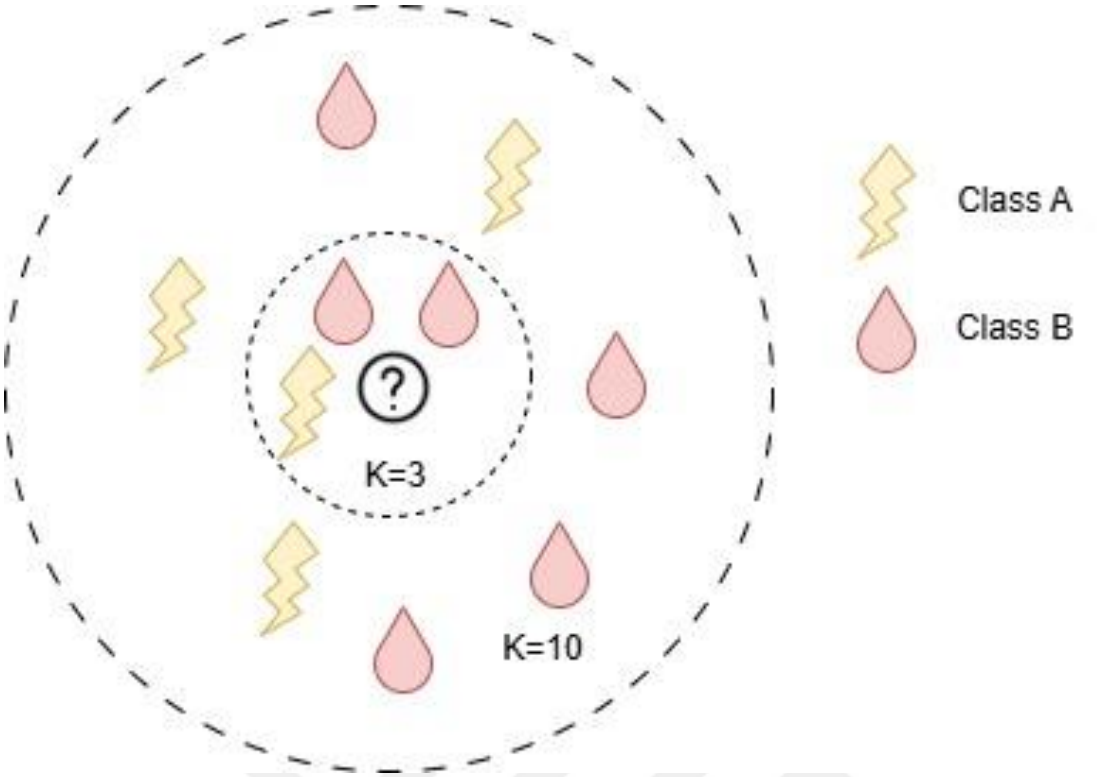

Figure 3.1: Overview of the K-Nearest Neighbor (KNN) method

Euclidean distance the direct distance between two points within a multi-dimensional space. Its prevalent application may stem from its straightforwardness and clarity. Grounded in the Pythagorean Theorem, it embodies the concept of the shortest route in two-dimensional space [35].

$$dEuclidean = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2} \tag{3.1}$$

Manhattan distance, namely L1 norm distance, serves a purpose in both spatial and mathematical frameworks. For two points, denoted as (p1) and (p2), in Euclidean n-space, the Manhattan distance is calculated by summing the lengths of the projections of the line segment connecting these points onto the respective coordinate axes. This distance quantifies the city block distance between two locations, considering only vertical and horizontal movements while traversing a grid-like geographical arrangement [36].

$$dManhattan = \sum_{i=1}^{d}|x_i - y_i| \tag{3.2}$$

Both the Euclidean and Manhattan distance measures are in fact special cases of the broader class of Minkowski distance measures, which are parameterized by $p \geq 1$ [37].

$$d\text{Minkowski} = \sum_{i=1}^{d} |x_i - y_i|^{p^{1/p}} \quad (3.3)$$

Selecting an appropriate value for K, referred to as K, is a significant factor that influences the efficiency of the KNN algorithm. This K parameter dictates the number of training samples involved in the decision-making process, with a larger K encompassing more neighbors. There are inherent trade-offs in the choice of K, as a smaller value can result in overfitting, whereas a larger value may cause under fitting. Methods such as cross-validation can be employed to assess a range of K values, facilitating a more informed selection. When analyzing models with K values greater than 5, it is essential to examine the overall shape of their decision boundaries, as the flexibility of KNN diminishes [38].

3.2 Decision Trees

A decision tree is a widely utilized algorithm in the fields of statistics and data analysis that facilitates the creation of a model to interpret how data can lead to specific conclusions. This approach employs a tree-structured model comprising decisions and potential outcomes. The structure is characterized by nodes, branches, and leaves. The nodes represent the various input features that are being analyzed. The branches signify the decision rules applied to derive predictions, while the leaves indicate the resultant outcomes of those predictions [39].

A decision tree is comprised of sequential binary splits that are selected to optimize information gain at each division, with "information" typically measured by Gini impurity or entropy. For classification tasks, the model designates the majority class, while for regression, the average value of the training instances present in a leaf is derived. A widely recognized limitation of decision trees is their propensity to over fit the training data. To mitigate this, one common technique is pruning, which involves applying a complexity parameter to eliminate splits that do not enhance the overall fit of the tree [40].

Decision tree divides data nodes recursively into two sub-nodes with a higher information gain than the previous node. To calculate the information gain, randomness in the data which is represented as entropy (S) is defined as in Equation 4.

$$\boldsymbol{S = \sum_{i=1}^{n} -p_i log(p_i)} \quad (3.4)$$

Where p_i is the probability of the i th event in the data given in Equation 5.

$$\boldsymbol{p_i = \frac{n_i}{|S|}} \tag{3.5}$$

where n_i is the particular class. From here on out information gain (IG) can be calculated by using Equation 6.

$$\boldsymbol{IG = S - \left(\sum_{v \in values(A)} \frac{|S_v|}{|S|}\right) Entropy\ (S_v)} \tag{3.6}$$

where S_v being the subset data after split and total subset S. The split location for each feature is determined with respect to highest information gain [41]. The process is summarized in Figure 3.2.

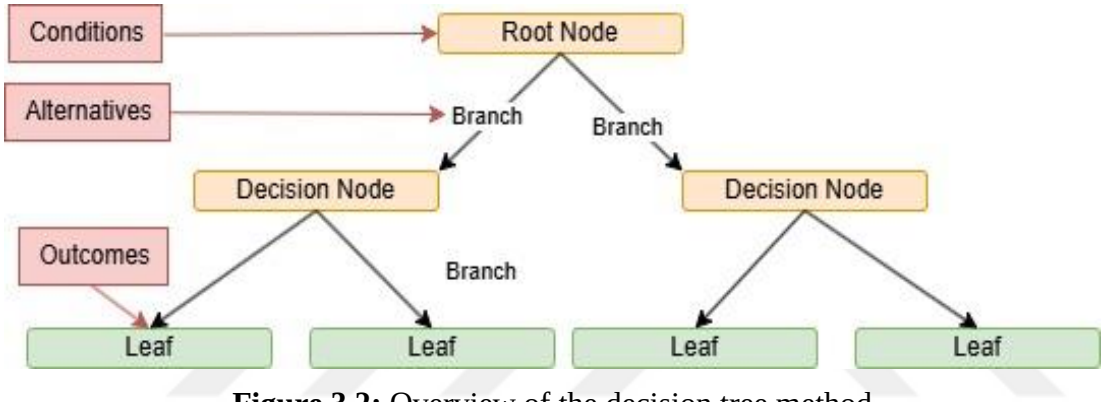

Figure 3.2: Overview of the decision tree method

3.3 Random Forest

A random forest classifier is used for classification utilizing ensemble learning. This method involves a collection of decision trees that work together to classify the most common designation for each input instance. Random Forest utilizes the principle of majority voting. Its mechanism generates a result according to the majority opinions of the classifier's votes. In parallel, predictions are made for each individual decision tree (see Figure 4). The ultimate prediction from the random forest is based on the prediction that receives the highest vote count from the decision trees. This classifier provides an effective method for discerning patterns within the input data [42].

A random forest is a classification algorithm that comprises a group of tree-structured classifiers, as represented in the equation 7.

$$\{h(x, z_k), k = 1,2..\} \tag{3.7}$$

In this context, z_k represents a collection of independent and identically distributed random vectors, and each decision tree provides a single vote for the most prevalent class corresponding to the input x as demonstrated in Figure 3.3 [43].

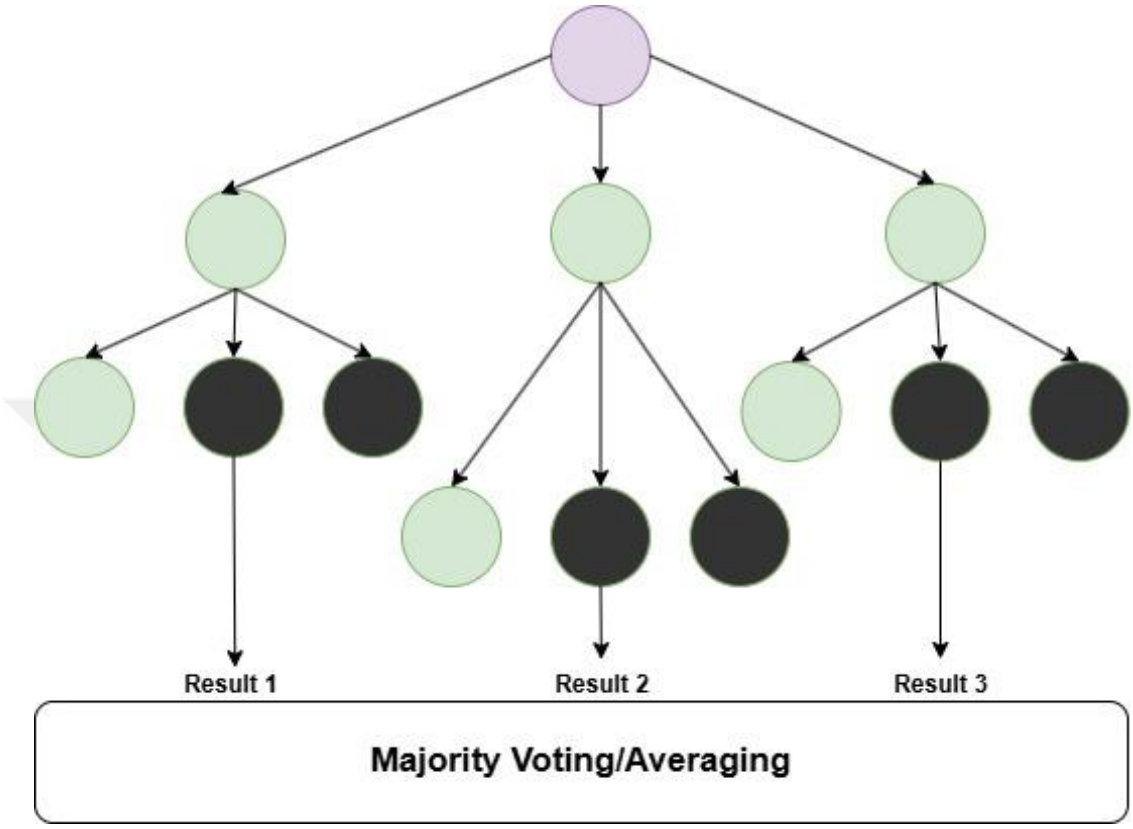

Figure 3.3: Overview of the Random Forest method

3.4 Gradient Boosting

Gradient Boosting is an ensemble learning technique utilized for tasks involving classification and regression. The method functions by training models in a sequential manner, with each new model designed to address the mistakes made by the previous one. In gradient boosting, every subsequent model is created with the purpose of decreasing the loss function, like mean squared error or cross-entropy, of the earlier model through gradient descent. At each iteration, the algorithm computes the gradient of the loss function based on the predictions and then trains an additional weak model to lower this gradient. The predictions from the new model are added to the ensemble, and this cycle continues until a specific stopping criterion is met, as illustrated in Figure 3.4 [44].

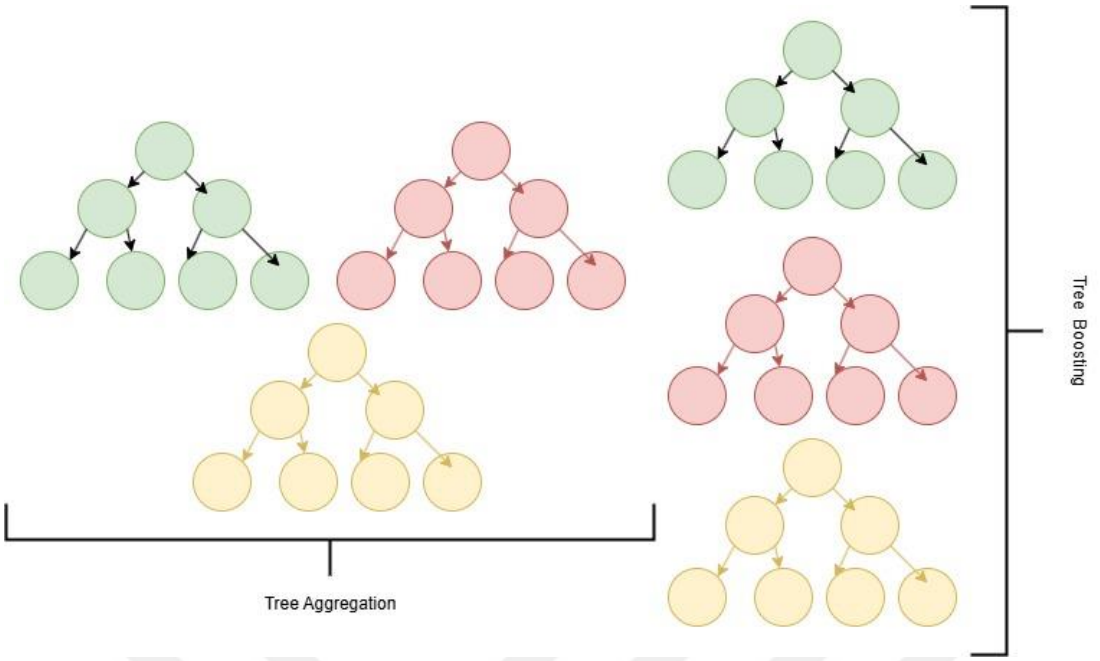

Figure 3.4: Overview of the Gradient Boosting method

The statistical component of boosting is clarified by presenting a boosting algorithm designed to enhance empirical risk through the application of steepest gradient descent in the function space [45].

The goal is to approximate the classification function $f(.)$ to accurately represent the association between the independent variable X and the dependent variable Y. In this context, the optimization challenge can be described in equation

$$f(.) = \arg min\{E_{y,Y}[\rho(Y, f(X))]\} \tag{3.8}$$

Where ρ is the loss function, leading to least square regression of the mean

$$f(x) = E(Y|X = x) \tag{3.9}$$

With observations $(y_1, x_1), \dots \dots, (y_n, x_n)$

$$f(.) = \arg min\left\{\frac{1}{n}\sum_{i=1}^{n}\rho(y_i, f(x_i))\right\} \tag{3.10}$$

3.5 Logistic regression

Logistic regression is a basic method used in ML, commonly applied to tasks involving binary classification. It serves as a bridge between simple linear regression and more advanced ML algorithms by providing a probabilistic framework to predict categorical

outcomes. Logistic regression is a classification method, distinguishing itself from regression through its primary objective of categorizing input into discrete classes based on training set. Logistic regression aims to represent the association between a dependent binary variable and one or multiple independent variables by employing the logistic function. This sigmoid function converts a linear aggregation of input features into a probability value that is limited to a range between zero and one. By doing so, logistic regression enables the assignment of data points into categories based on a given probability threshold, such as 0.5 for binary decisions [46, 47].

Provided a dataset that contains a distinctive set of features $f_s = (x_1, x_2, x_3, x_4, x_5, \dots x_i \dots x_n)$

The output response variable y_i for each input x_i take on a value of either zero or one. The response variable Y indicates the labeled class associated with the particular data sample. The logistic regression (LR) method calculates the probability that the data sample is classified into one of the two binary classes [48].

$$P(1|x, \alpha) = \frac{1}{1+e^{-(\alpha,x)}} \tag{3.11}$$

$$P(0|x, \alpha) = 1 - P(0|x, \alpha) \tag{3.12}$$

The linear equation representing the problem is expressed as $y = x\beta + \epsilon$, where y denotes the column vector of the response variable, x signifies the matrix of the dataset, β represents the parameter, and ϵ accounts for the error term. In this context, y is treated as a random variable that follows a probability distribution denoted as $P(y_i)$.

$$P(y_i) = \begin{cases} p_i & if y_i = 1 \\ 1 - p_i & if y_i = 0 \end{cases} \tag{3.13}$$

Logistic function;

$$E = [y_i = 1|x_i, \beta] = \frac{e^{x_i\beta}}{1+e^{x_i\beta}} \; for \; i = 1,2,3, \dots, n \tag{3.14}$$

Flow for logistic function classifier is provided in Figure 3.5.

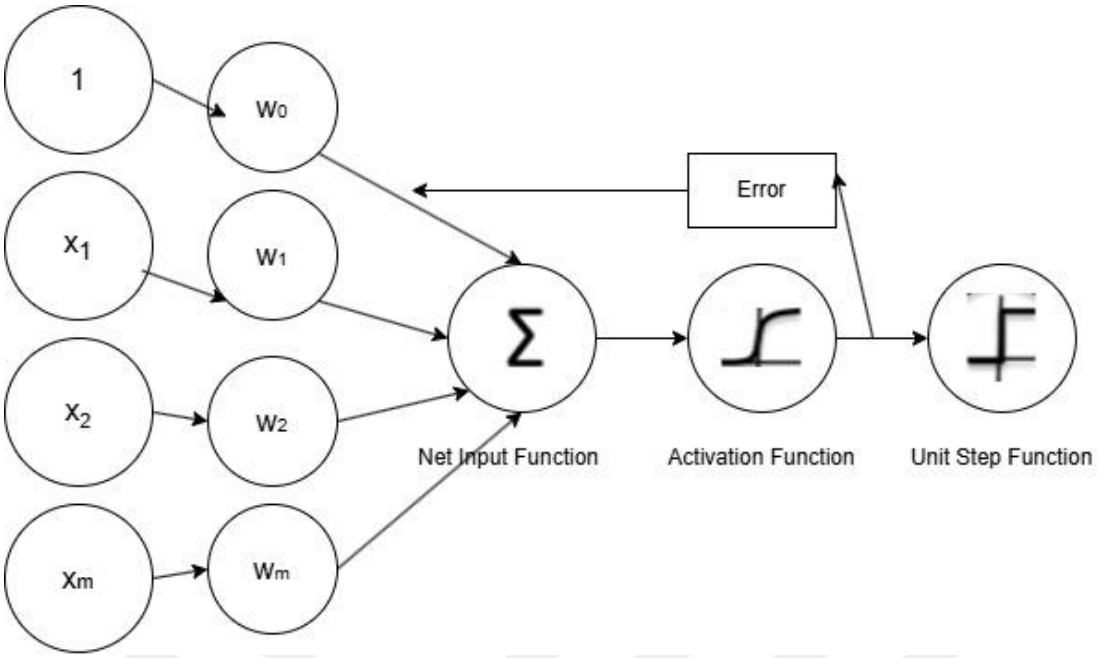

Figure 3.5. Overview of the Logistic Regression method

3.6 Support Vector Machine

Support Vector Machines (SVMs) are a versatile and powerful tool among the alternatives of supervised ML. SVMs have gained prominence due to their excellence in handling linearly and non-linearly separable data. They are especially recognized for their efficacy in high-dimensional environments, making them suitable for scenarios in which the quantity of features surpasses the number of available data samples. At their core, SVMs operate by constructing hyperplanes in a multidimensional space that optimally separate different classes of data points. The objective is to identify a hyper plane that maximizes the distance between two classes, thereby improving the model's capacity to generalize to new, unseen data, as illustrated in Figure 3.6 [49].

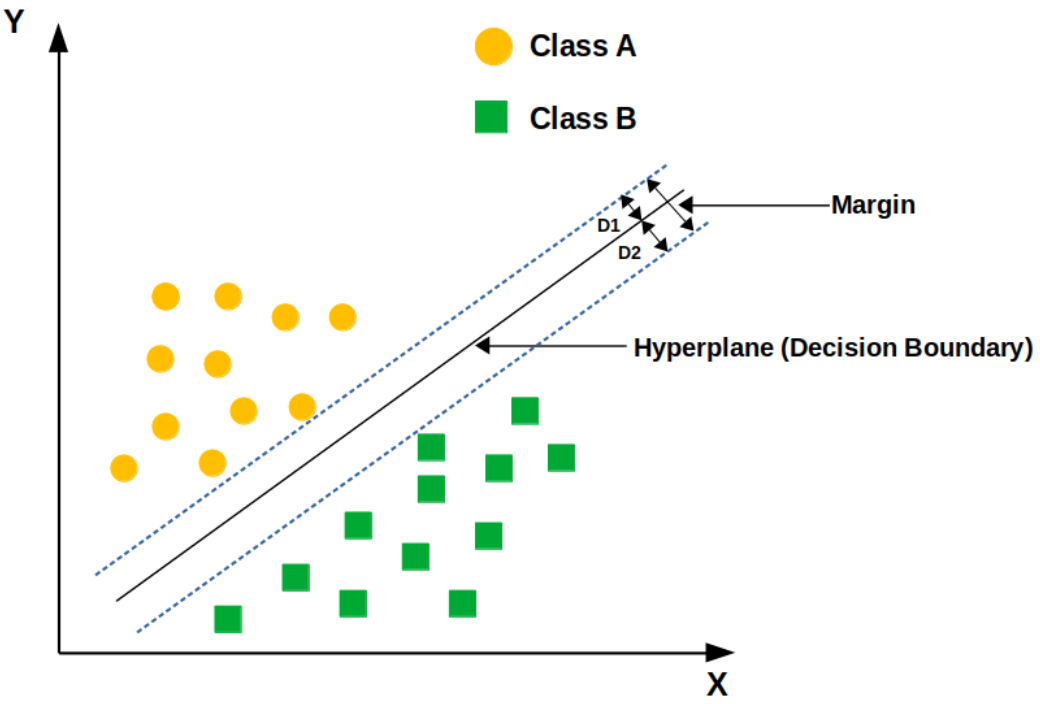

Figure 3.6: Overview of the SVM method

An intrinsic part of SVMs is the kernel trick, a mathematical technique that enables SVMs to classify data that cannot be separated by a straight line, by mapping it to more dimensional spaces. The prevalent kernels used in SVMs are linear, polynomial, radial basis function, and sigmoid. Each of these kernels is designed to address various types of non-linear patterns present in the data. While SVMs primarily perform well in binary classification scenarios, they can be modified for multi-class classification by employing techniques such as one-vs-one or one-vs-all methods. Additionally, SVMs are robust against overfitting, mainly due to the regularization parameter. This enables the adjustment of the balance between attaining a greater margin and reducing classification mistakes on the training dataset [50].

Considering a binary classification case

$$\{(x_i, y_i), i = 1, \dots, n, x \epsilon R^m, y \epsilon \pm 1\} \tag{3.15}$$

x_i represents m-dimensional input vector, y_i represents the category of x_i, defining the hyper plane

$$w.x + b = 0 \tag{3.16}$$

Provided w is weight factor and b represents bias, while satisfying

$$y_i[w.x_i] + b \geq 1 \tag{3.17}$$

Granting optimal hyper plane with

$$\min \varphi(w) = \frac{1}{2}\left||w|\right|^2 \tag{3.18}$$

Lagrange is applied to solve the equation

$$L(w, b, a) = \frac{1}{2}\left||w|\right|^2 - \sum_{i=1}^{n} a_i y_i (w.x_i + b) + \sum_{i=1}^{n} a_i \tag{3.19}$$

$a_i \geq 0$ Representing the Lagrange Multiplier, forming the decision function [50].

$$f(x) = (\sum_{i=1}^{n} a_i y_i (x_i * x + b)) \tag{3.20}$$

3.7 Multilayer Perceptron

The architecture of MLPs plays a crucial role in enabling these networks to model complex systems and processes. A MLP consists of several layers, each designed to process inputs and pass them to subsequent layers. The basic building block, the neuron, is inspired by biological neurons and is primarily defined by its activation function, which determines the neuron's output given a set of inputs. Typical activation functions comprise the sigmoid, hyperbolic tangent, and rectified linear unit. Each function introduces non-linearity into the network, enabling it to learn and model sophisticated patterns and relationships that are typical in nuclear processes where linear systems often fall short [51,52]. MLP is generally arranged with an input layer, one or more hidden layers, and an output layer. Each layer serves a distinct purpose: the input layer absorbs raw data, hidden layers transform the data, and the output layer produces the result. The depth and width of hidden layers are significant parameters that impact the network's capacity to learn. More hidden layers can capture complex feature hierarchies, whereas wider layers may allow for broader feature representation. This structural flexibility is critical in nuclear engineering applications, such as predicting reactor behavior or optimizing control processes, where the problems are characterized by vast input dimensions and require precise modeling [53, 54]. Training a MLP involves adjusting the weights of connections between neurons, using algorithms such as back propagation paired with an optimization technique like stochastic gradient descent. These weights determine how strongly a neuron's output influences the subsequent layer. During training, the network learns by minimizing the error between the predicted and actual outputs, iteratively refining the weights to improve performance represented in figure 3.7 [55, 56]. A neuron can be conceptualized as a computational unit that receives multiple input signals, processes

them, and outputs a single value. This process is determined by a weighted sum of the inputs, often integrated with a bias term, which represents an affine transformation. The significance of this transformation is magnified by the application of an activation function, which introduces non-linearity into the model. Without such non-linear characteristics, the capacity of MLPs to solve complex problems would be severely limited. Activation functions are pivotal in shaping the dynamics of neural networks, dictating how the transformed inputs are propagated through layers of neurons. There are several types of activation functions, each with distinct properties tailored to specific problem domains. The sigmoid function, defined by its S-shaped curve, compresses values between zero and one, making it suitable for scenarios requiring probability estimations. The hyperbolic tangent, which maps inputs to a range between -1 and 1, offers a more symmetric output, proving advantageous in certain types of data normalization. ReLU, distinguished by its linear response to positive inputs and saturation at zero for negative inputs, has become immensely popular because of its computational effectiveness and capacity to address the vanishing gradient issue that impacts deep neural networks [57].

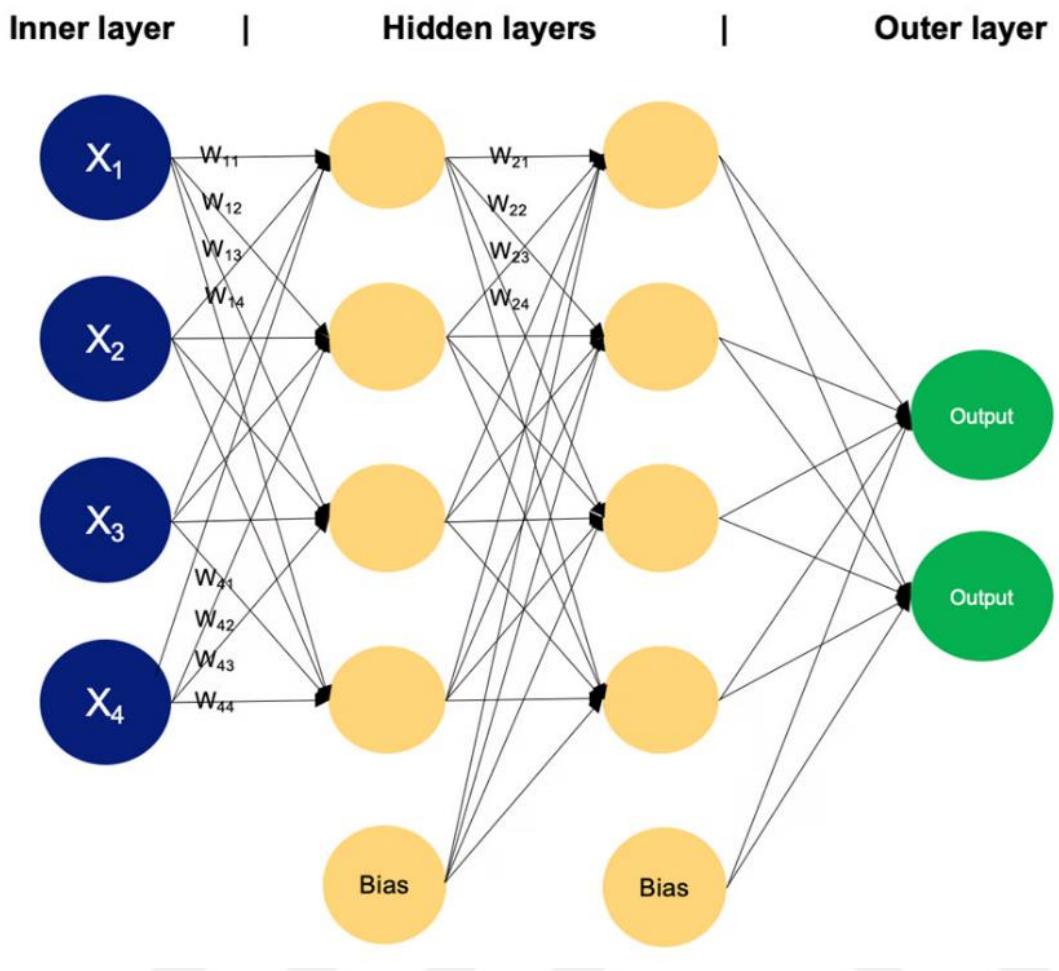

Figure 3.7: Overview of the Multilayer Perceptron method

3.8 Naïve Bayes

Naive Bayes methods consist of a collection of supervised learning algorithms that utilize Bayes' theorem under the assumption of conditional independence among each pair of features, conditional on the class variable's value [58]. Bayes' theorem defines the relationship between the class variable y and the dependent feature vector from x_1 to x_n as follows:

$$P(y|x_1, \ldots . . x_n) = \frac{P(y)P(x_1, \ldots . . x_n|y)}{P(x_1, \ldots . . x_n)} \quad (3.21)$$

Utilizing naïve conditional independence

$$P(x_i|y, x_1, \ldots . . x_{i-1}, x_{i+1}, \ldots . , x_n) = P(x_i|y) \quad (3.22)$$

Product for all i

$$P(y|x_1, \ldots . . x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, \ldots . . x_n)} \quad (3.23)$$

Considering $P(x_1, \ldots . . x_n)$ is constant, with classification rule

$$P(y|x_1, \ldots . . x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y) \quad (3.24)$$

$$\hat{y} = \arg max\, P(y) \prod_{i=1}^{n} P(x_i|y) \quad (3.25)$$

Maximum A Posteriori (MAP) estimation can be employed to estimate P(y) and $P(x_i|y)$, where P(y) represents the relative frequency of class y within the training set [59].

4. DATA GENERATION FOR MACHINE LEARNING

ML applications require large amounts of training and test data to produce models automatically. The data is used to generate models capable of making decisions automatically (e.g., classifications, clustering, or predictions) directly from the input data provided. Many widely used data-driven algorithms, such as SVMs, random forests, or gradient boosting require numerical inputs and do not handle missing, invalid, or infinite values properly. Preparing data for correct AI model training is often as challenging and time-consuming as developing and refining the model itself. In fact, data scientists reportedly spent about 80% of their time on data cleaning, model training, and tuning [60]. Despite being a crucial step in creating reliable ML models, the preparation of suitable data sets is neglected in almost 60% of ML studies [61]. Incomplete, incorrect, or irrelevant training and test data can lead to unreliable models, which in turn can make poor decisions. Modern AI applications are often used in programs capable of making decisions automatically. Trustworthy AI applications require high-quality training and test data, including many dimensions of data quality (e.g., accuracy, completeness, or consistency).

4.1 ASYST Code

The precise forecasting of thermal-hydraulic processes plays a critical role in ensuring safety across various engineering systems and technologies. As a result, it constitutes a significant area of active research and development in multiple industry sectors, including power engineering, automotive, environmental, and food industries, among others. Concerns regarding safety and efficiency are particularly paramount in the nuclear energy sector. Numerous issues are currently being addressed, such as the design of new reactors, extensions of service life, refurbishments of existing NPPs, the development of new fuel elements, and methods for waste disposal, all of which require a thorough analysis of thermal-hydraulic behavior. Many of these challenges are not solvable through analytical means; consequently, advanced computer software has been created. In response to this, a versatile simulation platform called ASYST

was introduced in 2007, resulting from a collaboration between AGH University of Science and Technology and Drives and Mining Controls. ASYST – A Graphic User Interface for RELAP5 has been developed by Australia Simulator Technologies Pty. The modeling capabilities of ASYST is extensive, enabling users to simulate systems ranging from simple to complex scenarios [62]. Building on RELAP5 as its core, ASYST incorporates various models that have been developed internationally. These diverse physical models can be integrated to tailor the simulation, allowing ASYST to be utilized for the analysis of Passage of Coolant Heat Exchangers (PCHX) as well as real incidents such as LOCA. While modeling Horizontal PCHX and Combined PCHX on simpler PCTRAN platforms can be complex, this particular test case was seamlessly executed in ASYST. The passage flow is independently calculated from junction flow using proximity algebra, which facilitates the setting of different levels of mixing modeling and comparisons to original code simplifications available on more accessible platforms. Furthermore, configuration testing has shed light on the impacts of various configurations on PCHX and has yielded sensitivity analyses along with recommendations for potential enhancements. To establish reliability, ASYST has been validated against experimental data, making it suitable for both academic research and industrial applications [63].

4.2 Transient Simulations

A transient is a change of reactor state from the steady state that may result in a change in coolant temperature, pressure, reactor power etc. [64, 65]. As such, to simulate these changes with respect to associated scenarios, it is necessary to first secure steady-state simulation of VVER-1000 reactors. All transient case simulations have to be achieved with manipulation of steady-state input file with trip and/or reactivity cards which adheres to the nodalization figure provided in figure 4.1, 50 reactor components, which are vital to determine reactor's behavior, given in Table 4.1 were chosen to acquire operational data such as mass flow rate, pressure, temperature, volume saturation temperature for partial pressure, saturation temperature for total pressure, volume vapor fraction, reactor power, and water level, resulting in 91 features. These 91 features were used for training and validation of AI methods considered, detailed in table B.1 in Appendix B.

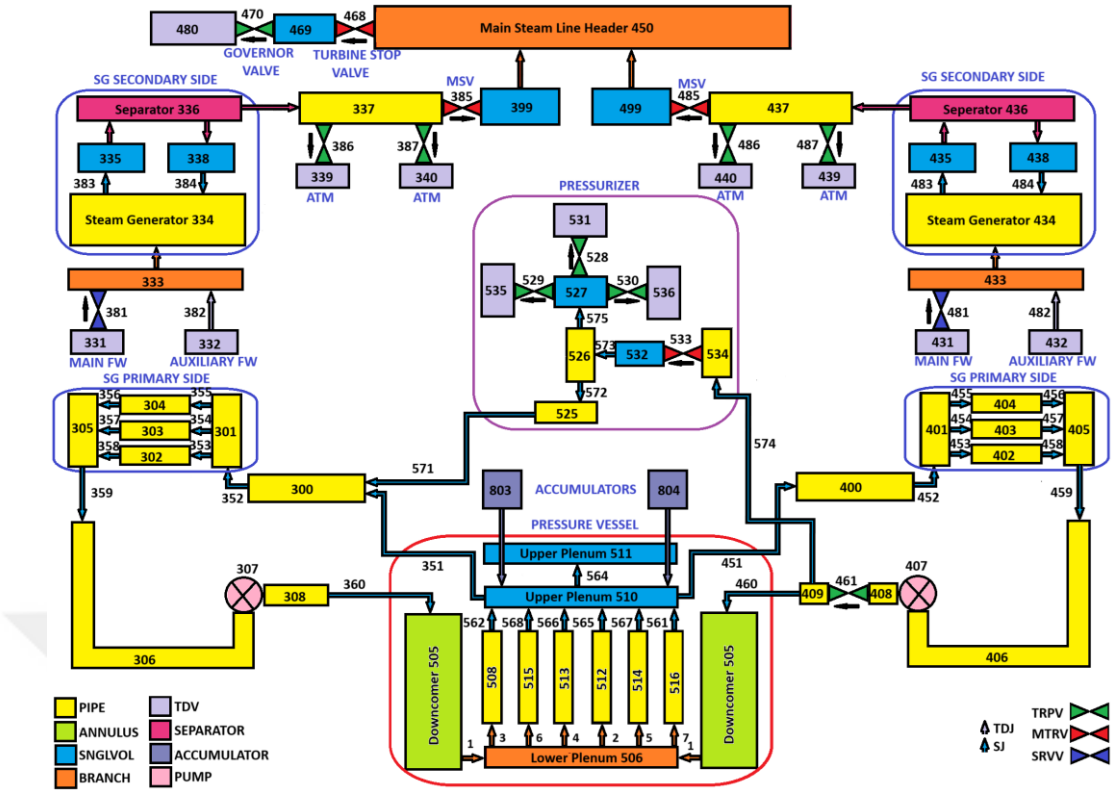

Figure 4.1: Nodalization of VVER-1000 power plant

Table 4.1. Hydrodynamic elements for nodalization

Element	Element
Reactor Power	Main Pump 1 - 4
Bypass Fuel Assembly (Hottest Channel)	Water Level in Steam Generator 1 - 4
Downcomer Annulus	Steam Generator 3 & 4 Outlet to Main Pump 3 & 4
Upper Plenum 1	Steam Line 1- 4 Safety Valve
Reactor Inlet Header 1 & 2 to Downcomer Annulus	Steam Generator 1 & 2 Secondary Side Inlet
Upper Plenum to Reactor Outlet Header 1	Steam Generator 3 & 4 Primary Side Outlet
Reactor Outlet Header on leg 1, 3, & 4	Turbine Stop Valve
Reactor Inlet Header on Leg 3 & 4	Main Steam Header Valve 3 & 4
hot leg 3 & 4 to Steam Generator 3 & 4	Feed Water Valve 1 - 4
Steam Generator 3 & 4 Lower Part	Pressurizer Relief Valve 1 - 3
Steam Generator 3 & 4 Medium Part	Water Level in Pressurizer
Steam Generator 3 & 4 Upper Part	Main Steam Header Valve 1

The input file has 160 active trip cards to simulate scenarios, 101 of which are variable and 59 logical. Logical trips bind other trips that can be logical or variable, to each other, while variable trips deal with physical properties of the system such as pressure, temperature, saturation temperature, flow rate, power and water level. Trip distribution for block and type is provided in Table 4.2.

Table 4.2: Trips for input file

Trip Block	Logical	Variable
General Trips	12	25
Main Steam Isolation Valve Trip		5
Reactor Coolant Pump Trip 1.Block	2	2
Reactor Coolant Pump Trip 2.Block	2	2
Reactor Coolant Pump Trip 3.Block	2	2
Reactor Coolant Pump Trip 4.Block	2	2
Engineering Safety Features Trips 1.Block		4
Engineering Safety Features Trips 2.Block		8
Engineering Safety Features Trips 3.Block		7
Engineering Safety Features Trips 4.Block	6	17
Reactor Trip 1.Block	3	
Reactor Trip 2.Block	3	
Reactor Trip 3.Block	3	
Reactor Trip 4.Block	3	
Reactor Trip 5.Block	3	2
Steam Generator 1 Primary Side Depressurization		2
Steam Generator 2 Primary Side Depressurization	3	4
Steam Generator 3 Primary Side Depressurization	3	5
Steam Generator 4 Primary Side Depressurization	3	4
Pressurizer 1.Block	6	6
Pressurizer 2.Block		4
Pressurizer 3.Block	3	1
Grand Total	**59**	**101**

For each transient sub-scenario, in order to achieve stable steady-state operation, 3500 seconds of steady-state operation was performed. The transients were initiated after 3500 s of simulation and SCRAM was activated 4 seconds after each transient commenced. At first, the steady state full power operation case was simulated to produce the reference case. This reference case was then used to generate transients. Five major transients; reactivity insertion with rod withdrawal, steam leak from pressurizer, loss of flow (LOFA), loss of coolant (LOCA) in hot leg, and LOCA in cold leg were modelled. For rod withdrawal transient, the control rod was assumed to

be fully in position at the beginning of the transient and rod withdrawal was performed with 5% increments and for LOCA hot and cold leg transients, the break size was increased with 5% increments while for steam leak from pressurizer and for loss of flow transients, the number of valves in pressurizer and pumps that are involved in the transient were increased one by one, resulting in a total of 53 sub-scenarios given in Table 4.3.

Table 4.3: Sub-Scenarios for the transients

Main Transient	Sub-Scenario	Main Transient	Sub-Scenario
Steady State	Steady State	LOCA Hot Leg	LOCAHotLeg55
Rod Withdrawal	RodWithdrawal05	LOCA Hot Leg	LOCAHotLeg60
Rod Withdrawal	RodWithdrawal10	LOCA Hot Leg	LOCAHotLeg65
Rod Withdrawal	RodWithdrawal15	LOCA Hot Leg	LOCAHotLeg70
Rod Withdrawal	RodWithdrawal20	LOCA Hot Leg	LOCAHotLeg75
Rod Withdrawal	RodWithdrawal25	LOCA Hot Leg	LOCAHotLeg80
Rod Withdrawal	RodWithdrawal30	LOCA Hot Leg	LOCAHotLeg90
Rod Withdrawal	RodWithdrawal35	LOCA Hot Leg	LOCAHotLeg85
Rod Withdrawal	RodWithdrawal40	LOCA Cold Leg	LOCAColdLeg05
Rod Withdrawal	RodWithdrawal45	LOCA Cold Leg	LOCAColdLeg10
Steam Leak from Pressurizer	PrezLoss1	LOCA Cold Leg	LOCAColdLeg15
Steam Leak from Pressurizer	PrezLoss12	LOCA Cold Leg	LOCAColdLeg20
Steam Leak from Pressurizer	PrezLoss123	LOCA Cold Leg	LOCAColdLeg25
Loss of Flow	LossofFlow4	LOCA Cold Leg	LOCAColdLeg30
Loss of Flow	LossofFlow34	LOCA Cold Leg	LOCAColdLeg35
Loss of Flow	LossofFlow234	LOCA Cold Leg	LOCAColdLeg40
Loss of Flow	LossofFlow1234	LOCA Cold Leg	LOCAColdLeg45
LOCA Hot Leg	LOCAHotLeg05	LOCA Cold Leg	LOCAColdLeg50
LOCA Hot Leg	LOCAHotLeg10	LOCA Cold Leg	LOCAColdLeg55
LOCA Hot Leg	LOCAHotLeg15	LOCA Cold Leg	LOCAColdLeg60
LOCA Hot Leg	LOCAHotLeg20	LOCA Cold Leg	LOCAColdLeg65
LOCA Hot Leg	LOCAHotLeg25	LOCA Cold Leg	LOCAColdLeg70
LOCA Hot Leg	LOCAHotLeg30	LOCA Cold Leg	LOCAColdLeg75
LOCA Hot Leg	LOCAHotLeg35	LOCA Cold Leg	LOCAColdLeg80
LOCA Hot Leg	LOCAHotLeg40	LOCA Cold Leg	LOCAColdLeg85
LOCA Hot Leg	LOCAHotLeg45	LOCA Cold Leg	LOCAColdLeg90
LOCA Hot Leg	LOCAHotLeg50		

4.2.1 Steady-state

In order to obtain the steady-state, all active trip cards have been assessed and modified in a way that their set points are not reached, thus rendering them inactive without compromising the unity of input file. For reactor to stabilize, 3500 seconds of simulation time were granted and stabilization of all parameters were observed for 91 minor edit parameters (for features described before). Graphical representation for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor, and steam generator water level for steady state sub-scenario can be found in Figure 4.2 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. Accumulators automatically kick in when pressure falls under 6079500 Pa and pressurizer works as a balancer.

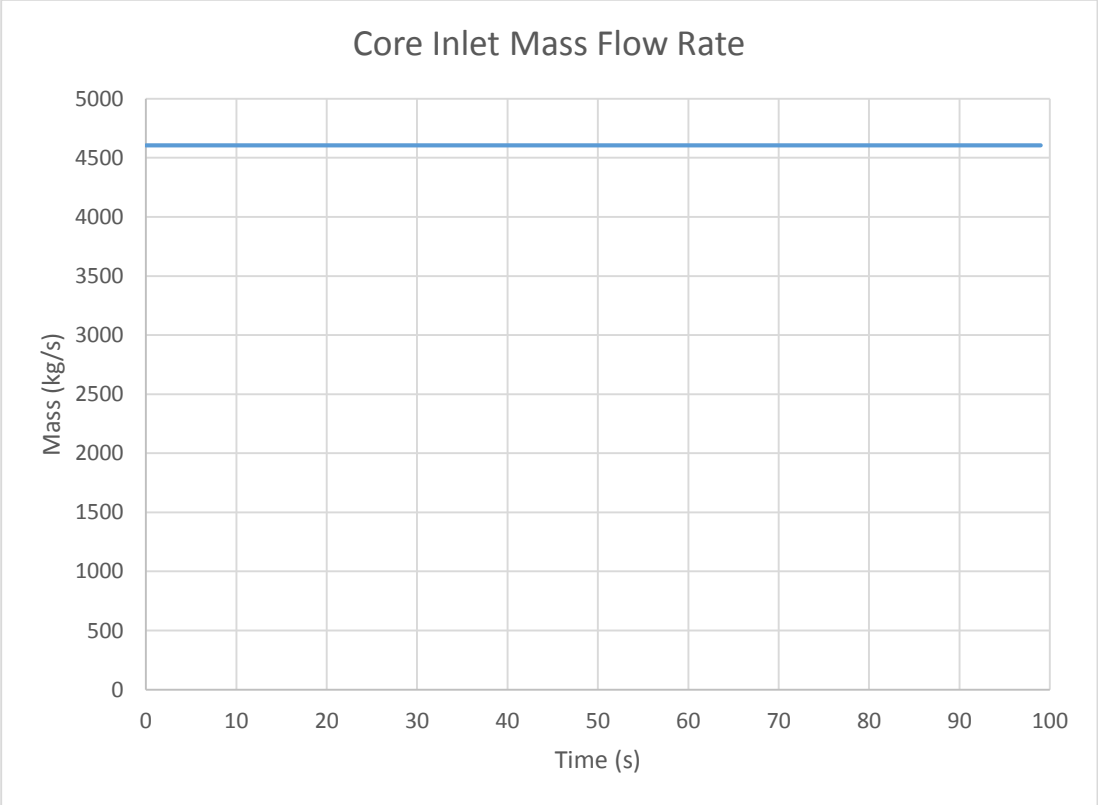

Figure 4.2.1: Core inlet mass flow rate vs time for steady state

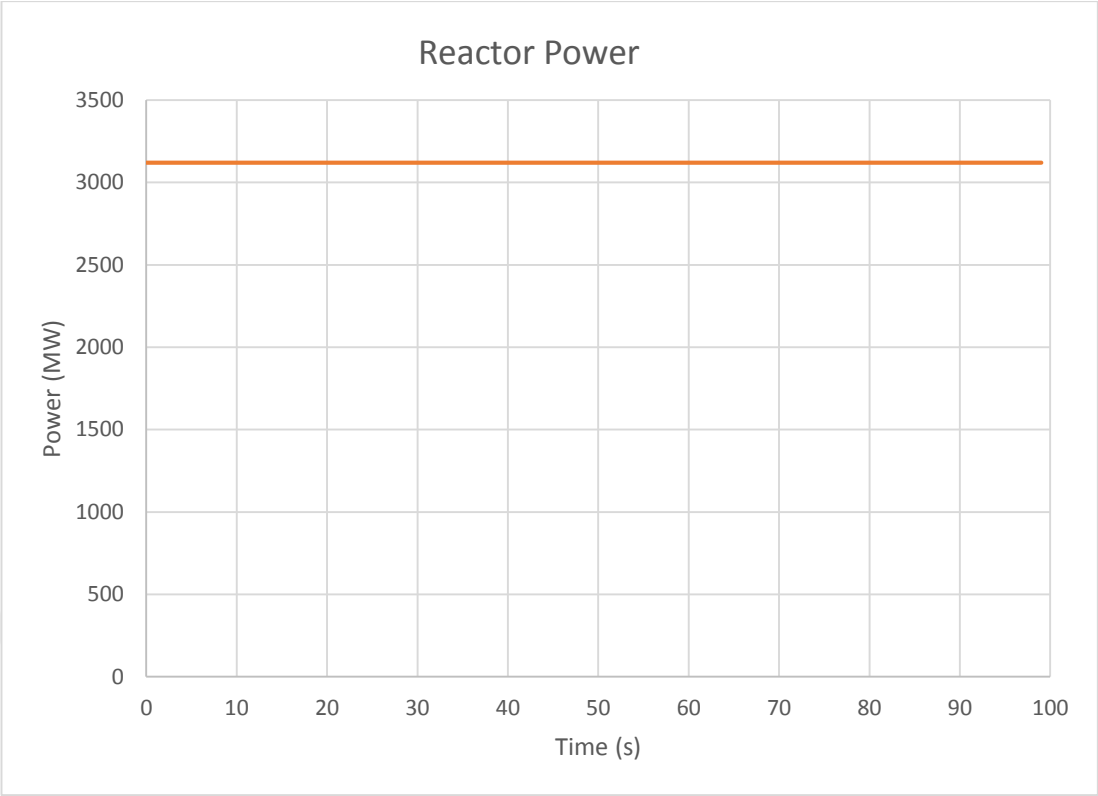

Figure 4.2.2: Reactor power vs time for steady state

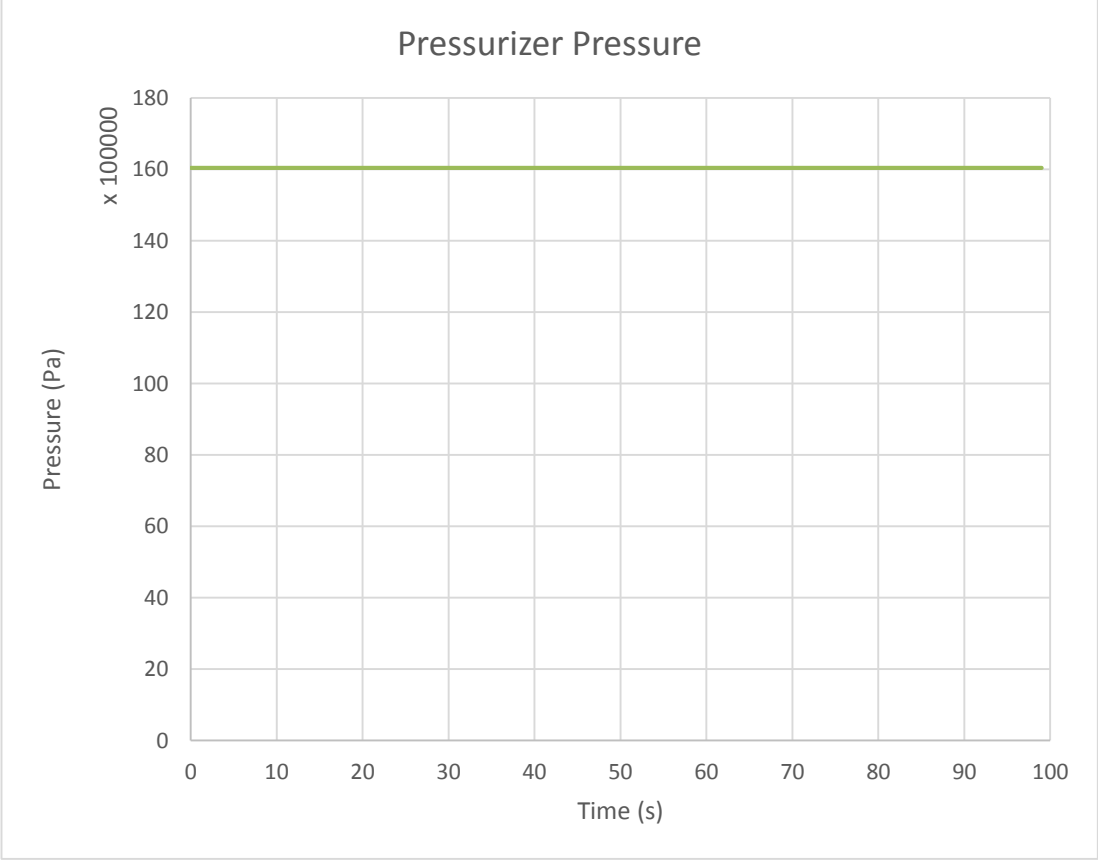

Figure 4.2.3: Pressurizer pressure vs time for steady state

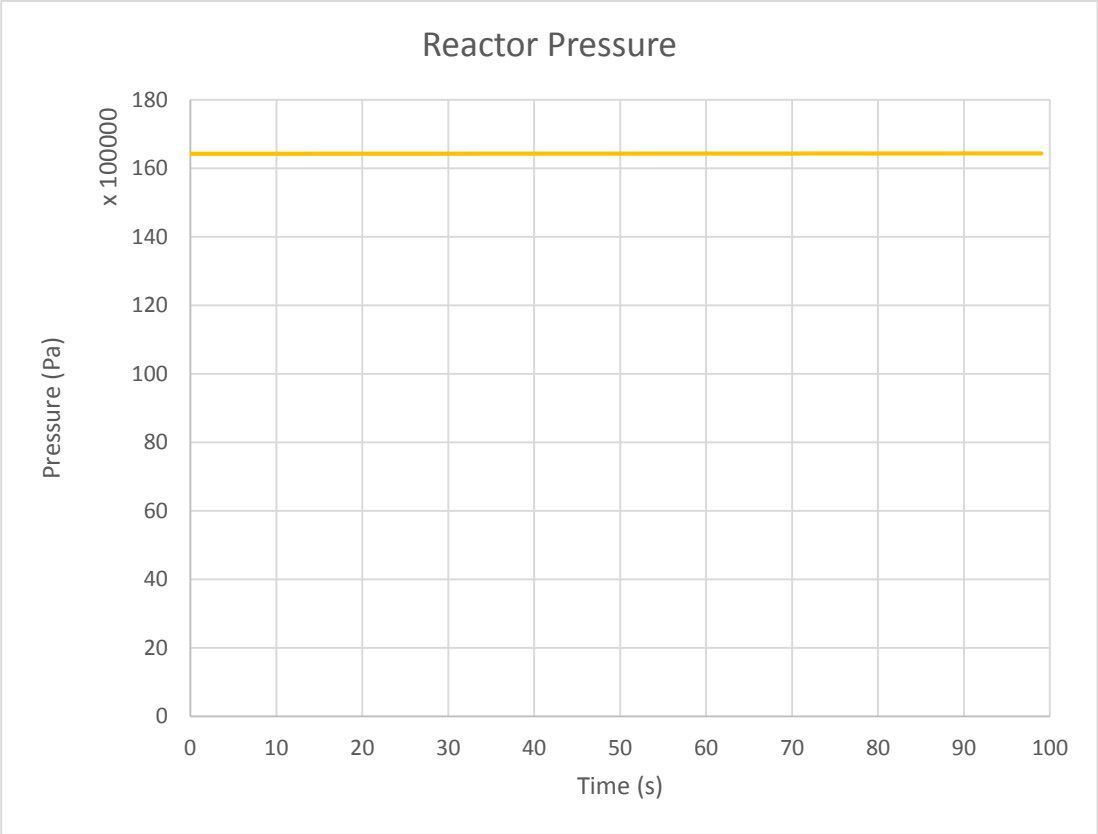

Figure 4.2.4: Reactor pressure vs time for steady state

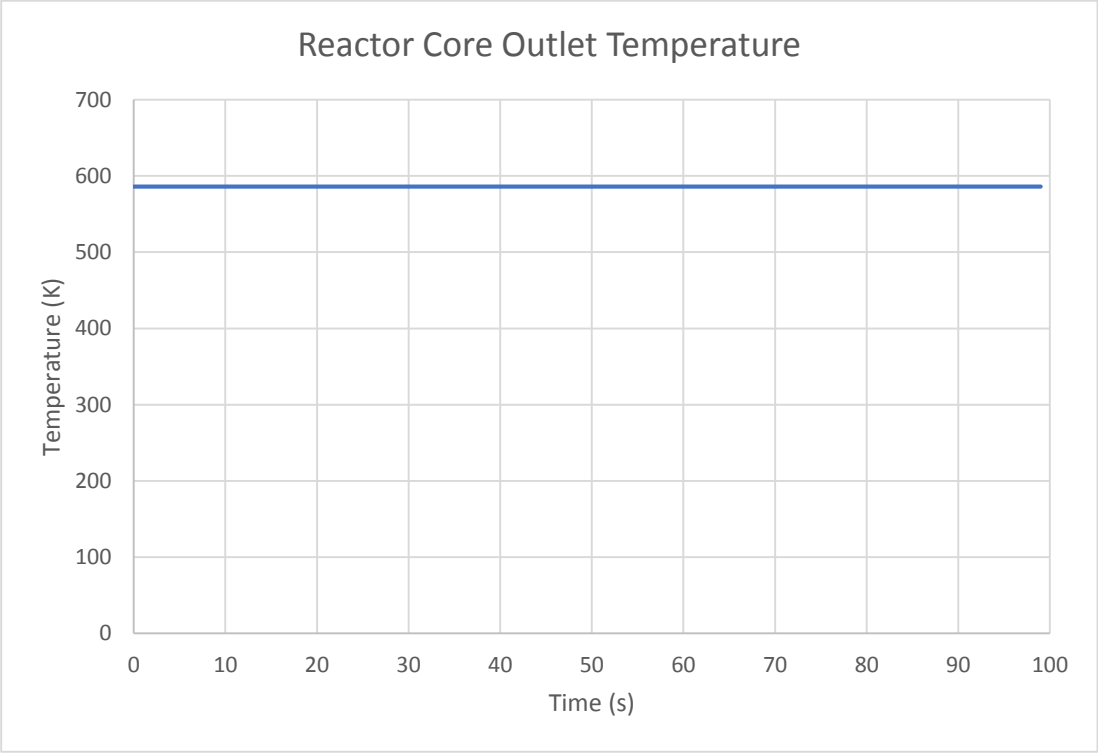

Figure 4.2.5: Reactor core outlet temperature vs time for steady state

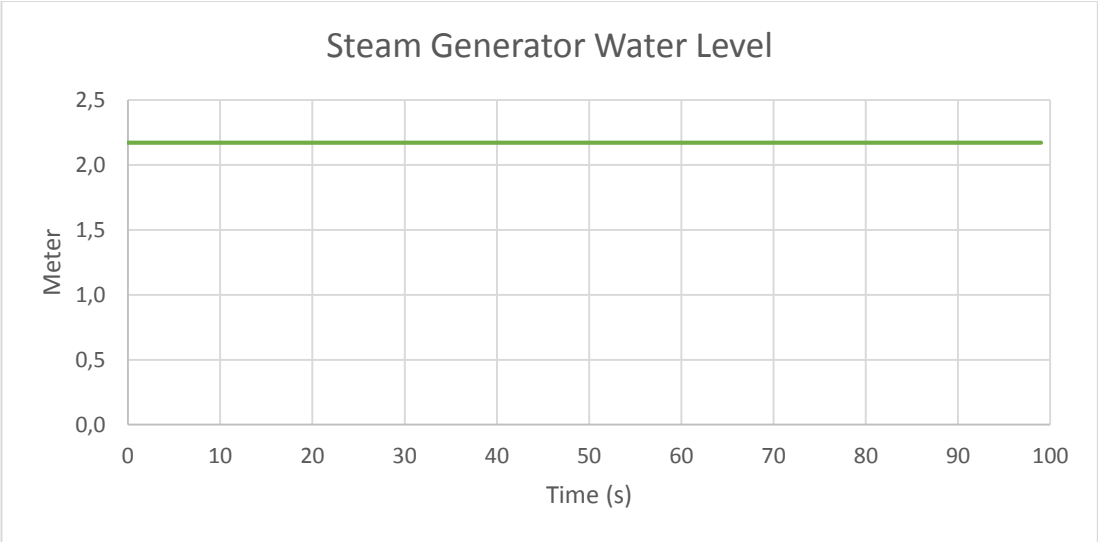

Figure 4.2.6 Steam generator water level vs time for steady state

4.2.2 Reactivity insertion with rod withdrawal transient

Rod withdrawal transients in PWRs are initiated when control rod assembly are withdrawn at a faster rate than that specified in the technical specification documents. These events are primarily caused by operator errors and occur infrequently [66, 67].

To simulate reactivity insertion, scram table card was manipulated for positive reactivity insertion for the first 4 seconds, followed by negative reactivity insertion due to SCRAM since nuclear reactors respond to transients with automated SCRAM which has 3 – 4 seconds response time.

The input segment modified for the transient is seen in Figure 4.3. The time trip 590 was arranged to initiate positive reactivity insertion after 3500 s of steady-state operation.

The level of the transient was determined by data cards 20200103 and 20200104, encapsulating the time frame between second 0 and 4 and 0.5 dollars of reactivity as a parameter. Nine sub-scenarios were produced by increasing reactivity 0.5 dollars for each scenario up to 4.5 dollars.

```
590  time      0          gt  null  0     3500.0          1
*SCRAM TABLE DATA CARD
20200100    reac-t  590
*
20200101  -1.0     00.0000000
20200102   0.0     00.0000000
20200103   0.0     00.4500000
20200104   4.0     00.4500000
20200105   4.1   -01.4912300
20200106   5.5   -12.7841300
20200107   5.75  -14.4876400
20200108   6.4   -15.9763000
20200109 10000.   -15.9763000
*20200110 10000.  -15.9763000
```

Figure 4.3.1: Scram table for rod withdrawal transient

Graphical representations for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor and steam generator water level for RodWithdrawal05 sub-scenario can be found in Figure 4.4 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. As expected, for rod withdrawal sub-scenario, power profile shows an increase for 4 seconds and then displays a steep decrease as SCRAM kicks in. After 4th second the pressurizer kicks in to support reactor, resulting in a minor increase mass flow in the core, while pressure of pressurizer and reactor decreases slightly and core temperature increases for 4 seconds and decreases to 535 K in accordance with SCRAM with power production.

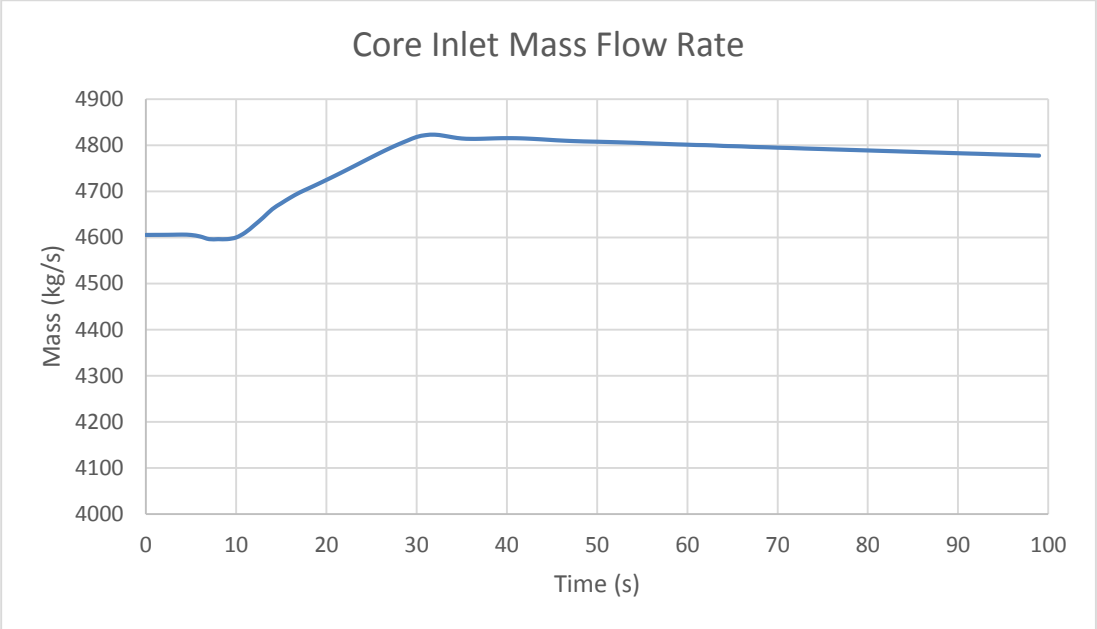

Figure 4.3.2: Core inlet mass flow tate vs time for for rod withdrawal transient

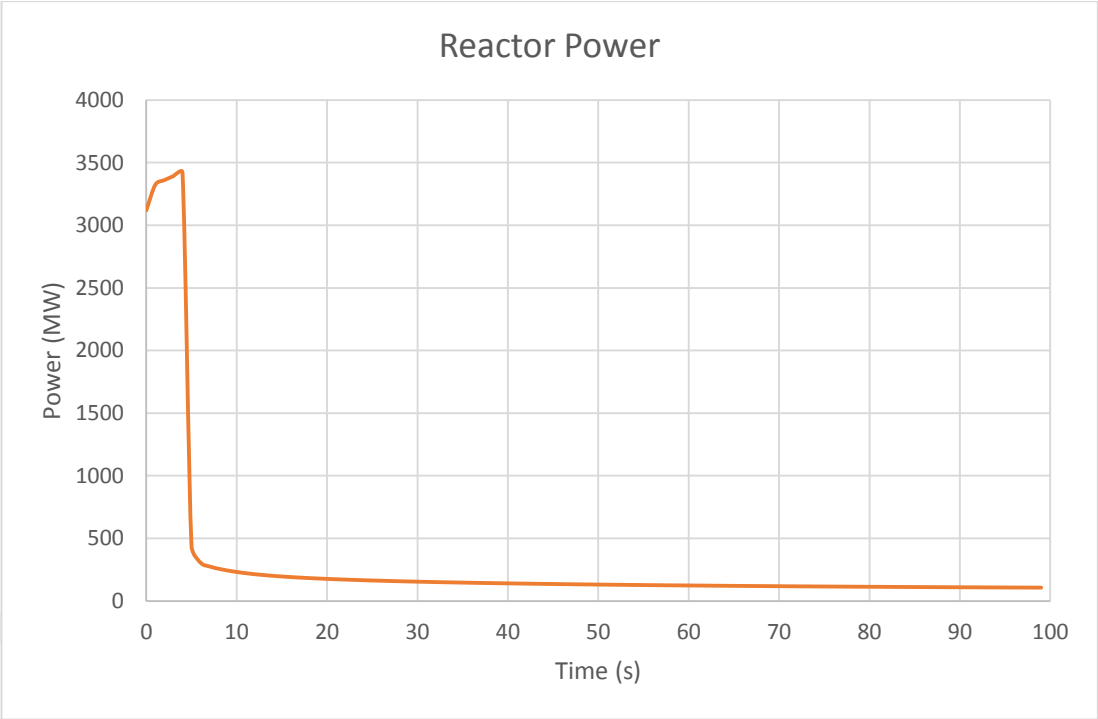

Figure 4.3.3: Reactor power vs time for rod withdrawal transient

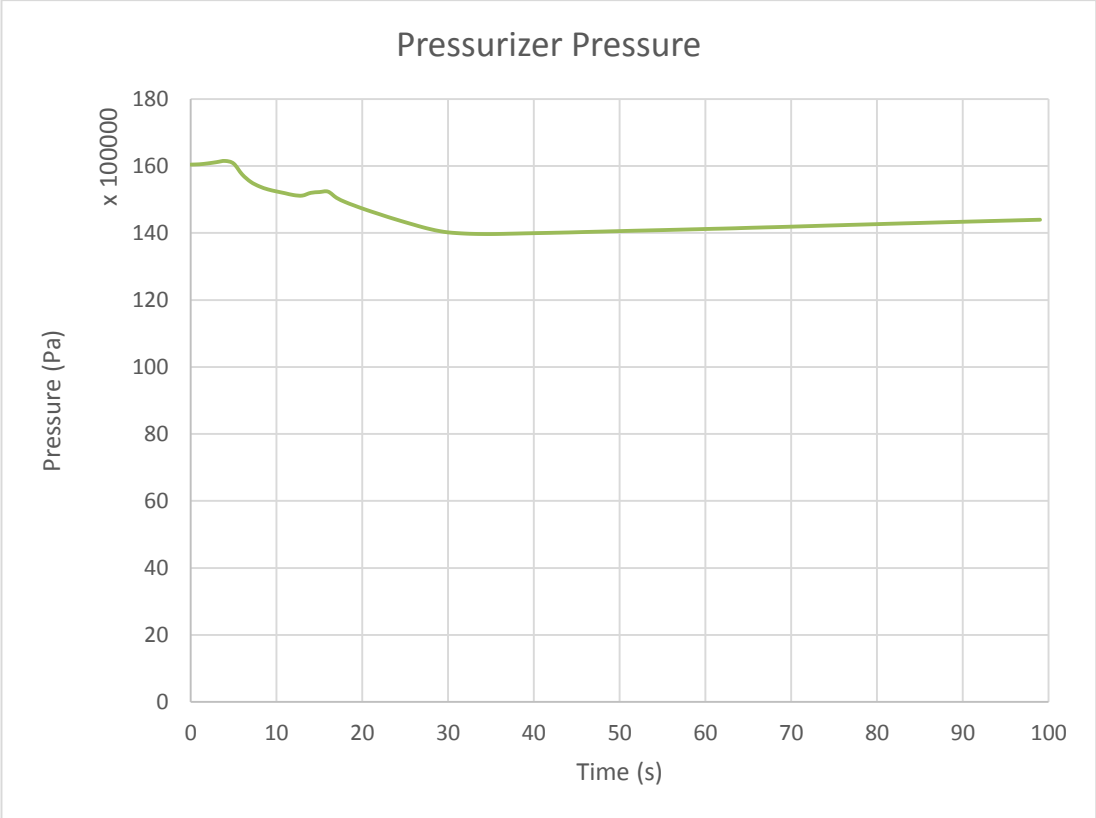

Figure 4.3.3: Pressurizer pressure vs time for rod withdrawal transient

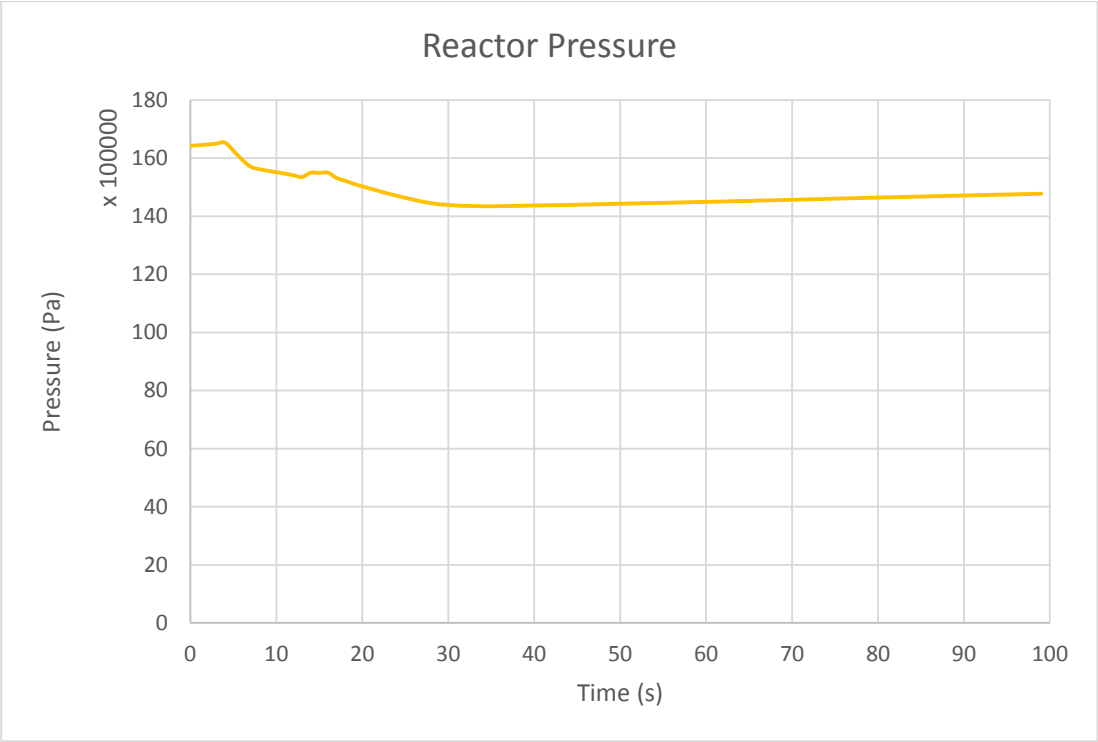

Figure 4.3.4: Reactor pressure vs time for rod withdrawal transient

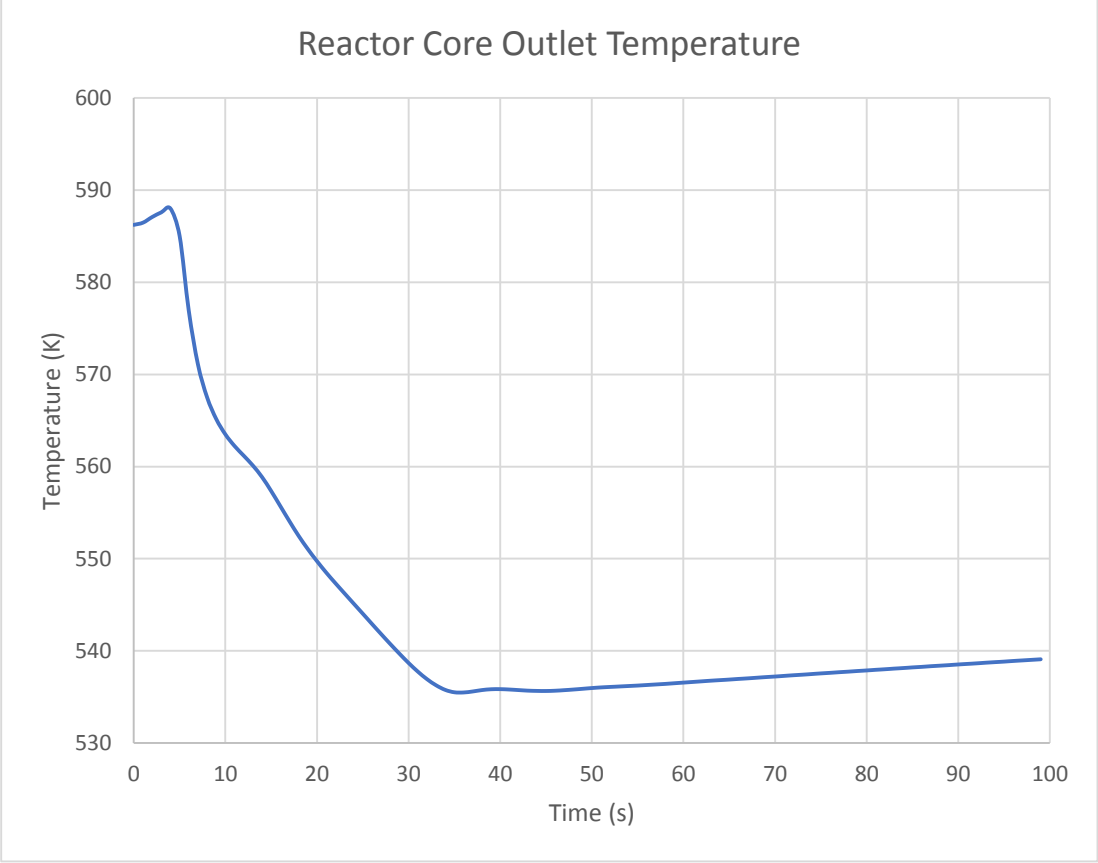

Figure 4.3.5: Reactor core outlet tempertature vs time for rod withdrawal transient

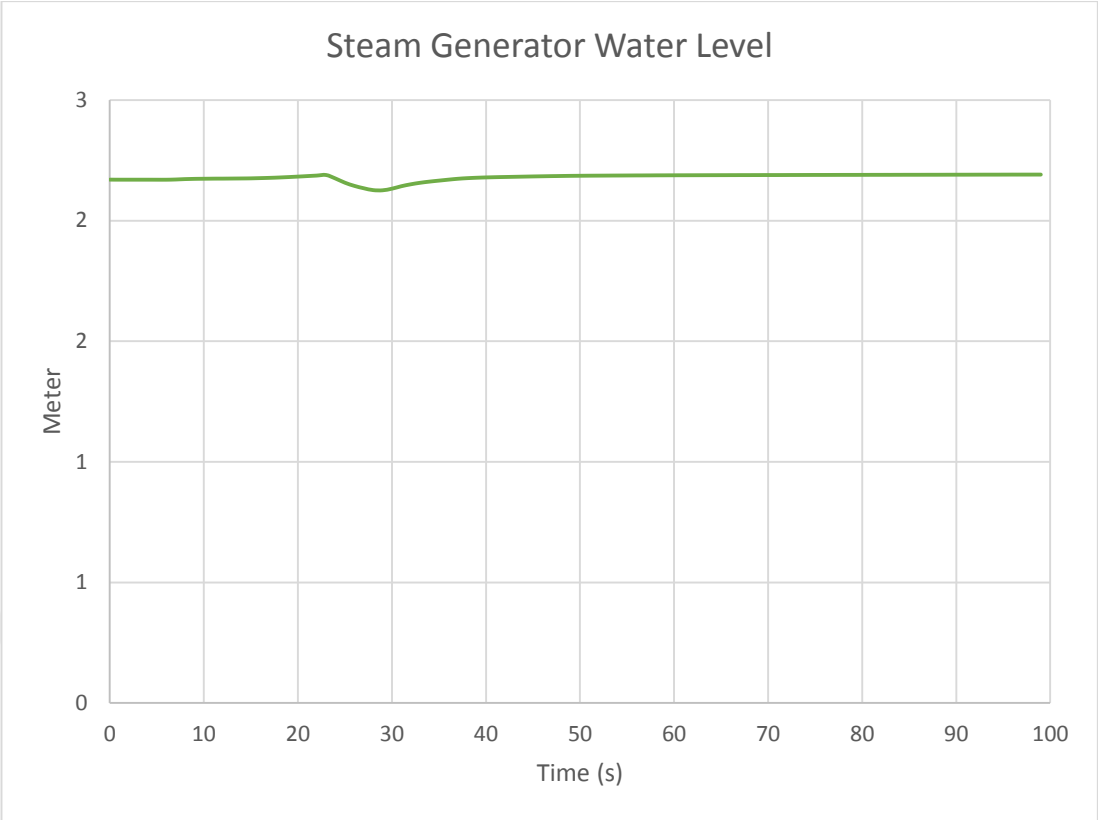

Figure 4.3.6: Steam generator water level vs time for withdrawal transient

4.2.3 Steam leak from pressurizer transient

The pressurizer is a vertical cylindrical vessel filled with water, heated by electrical heaters/steam lines. Its function is to continuously monitor and control the pressure variations in the primary loop, to control or compensate for the variations of mass and volume of water in the primary water circuit during reactor operation, and to maintain the water in the reactor in liquid state even at high temperature. It is a critical component in a PWRs and is a part of the primary cooling system having water under pressure [68]. To simulate the Steam Leak from Pressurizer scenario, trip valves with element number 528, 529, and 530 which are connected to element 527 that is in the pressurizer block in nodalization Figure 4.5, were manipulated with a time trip 419 that initiates at second 3500 of the simulation, causing the trip valves to open and hence simulate the leak from pressurizer. Since trip valves operate with a binary perspective, that is “open” or “closed”, three sub-scenarios have been assessed, with only 1, 2 and then respectively all 3 have been opened.

```
*------------------------------------------------------------------------
*                  TRIP CARDS
*------------------------------------------------------------------------
*500   time  0   gt    null        0      5.0       1
419  time     0          gt   null    0      3500.       n
500      p   134010000    gt  null 0  5.6e+06   n  -1.0
499      p   134010000    gt   null  0  7.15e+06 n  -1.0
650   500  and  651   n    -1.0

*-------------------------------------------------------------------*
5280000 trpvlv valve
5280101 527010002 531000000  0.00165  0.0  0.0  0000120 1.0 1.0 1.0
5280201 1 0.0 0.0 0.0
5280300   trpvlv
5280301   419    *578
*-------------------------------------------------------------------*
5290000 trpvlv valve
5290101 527010002 535000000  0.00165  0.0  0.0  0001120 1.0 1.0 1.0
5290201 1 0.0 0.0 0.0
5290300   trpvlv
5290301   666
*-------------------------------------------------------------------*
5300000 trpvlv valve
5300101 527010002 536000000 0.00165 0.0  0.0  0001120 1.0 1.0 1.0
5300201 1 0.0 0.0 0.0
5300300   trpvlv
5300301   666
*-------------------------------------------------------------------*
```

Figure 4.4.1: Trip card input for steam leak from pressurizer transient

Graphical representations for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor, steam generator water level and relief valve mass flow rate for PrezLoss123 sub-scenario can be found in Figure 4.6 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. As expected, mass flow for relief valve displays almost an instant increase and continues to flow although rate decreases in time of simulation. Pressurizers kick in at 4th second increasing Core Inlet mass flow rate, but due to strong loss of pressure from trip valves pressurizer and reactor pressure shows a strong decrease while outlet reactor temperature shows an expected fall due to SCRAM.

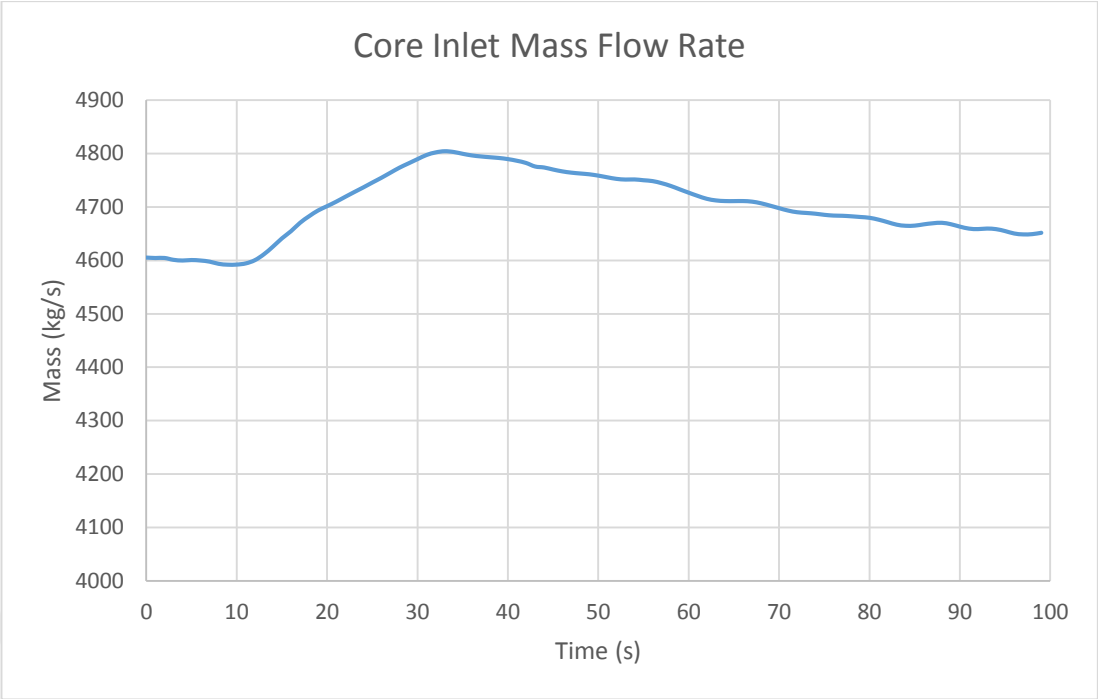

Figure 4.4.2: Core inlet mass flow rate vs time for steam leak from pressurizer transient

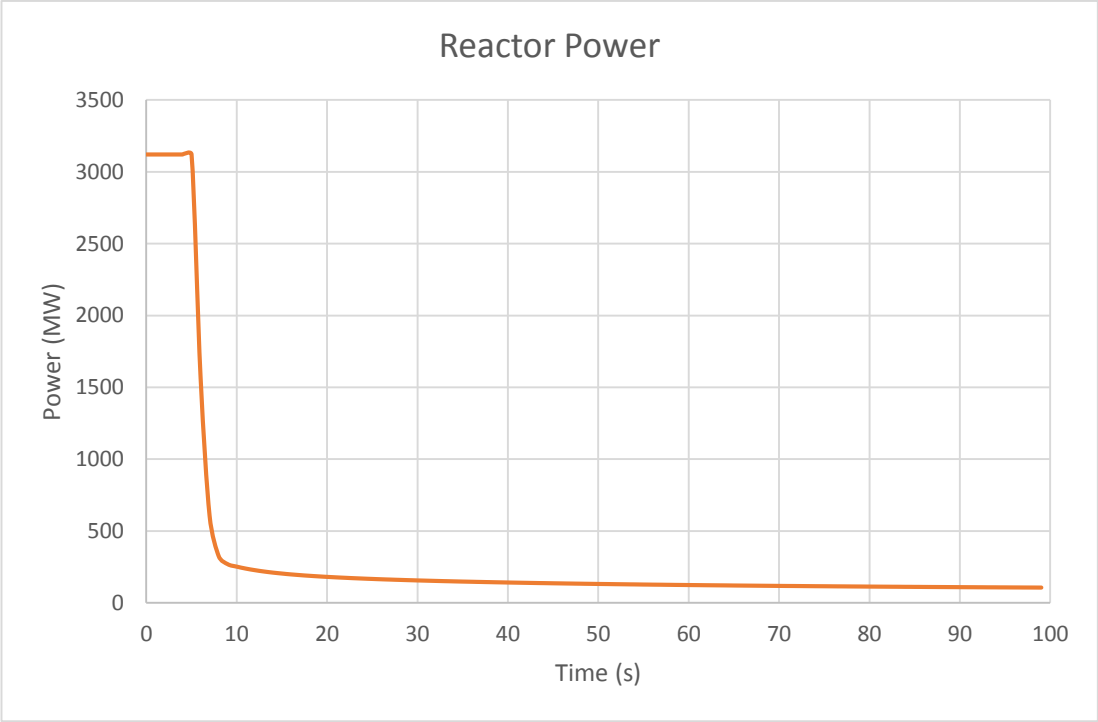

Figure 4.4.3: Reactor power vs time for steam leak from pressurizer transient

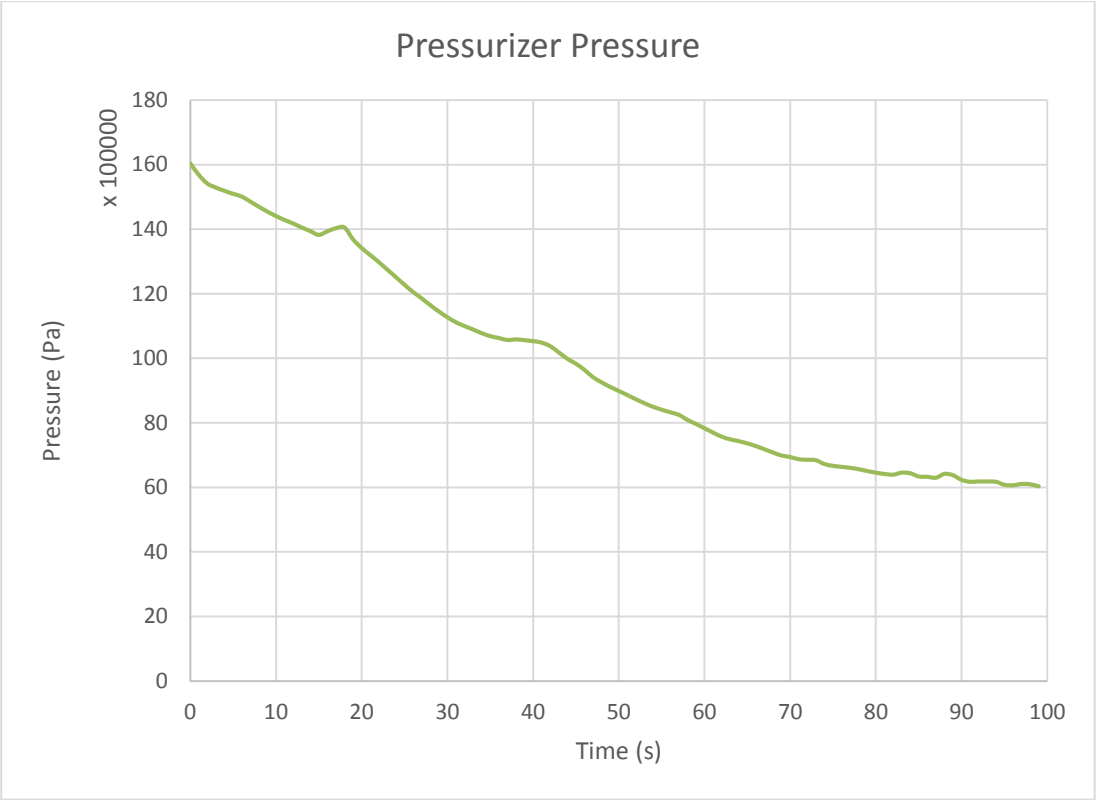

Figure 4.4.4: Pressurizer pressure vs time for steam leak from pressurizer transient

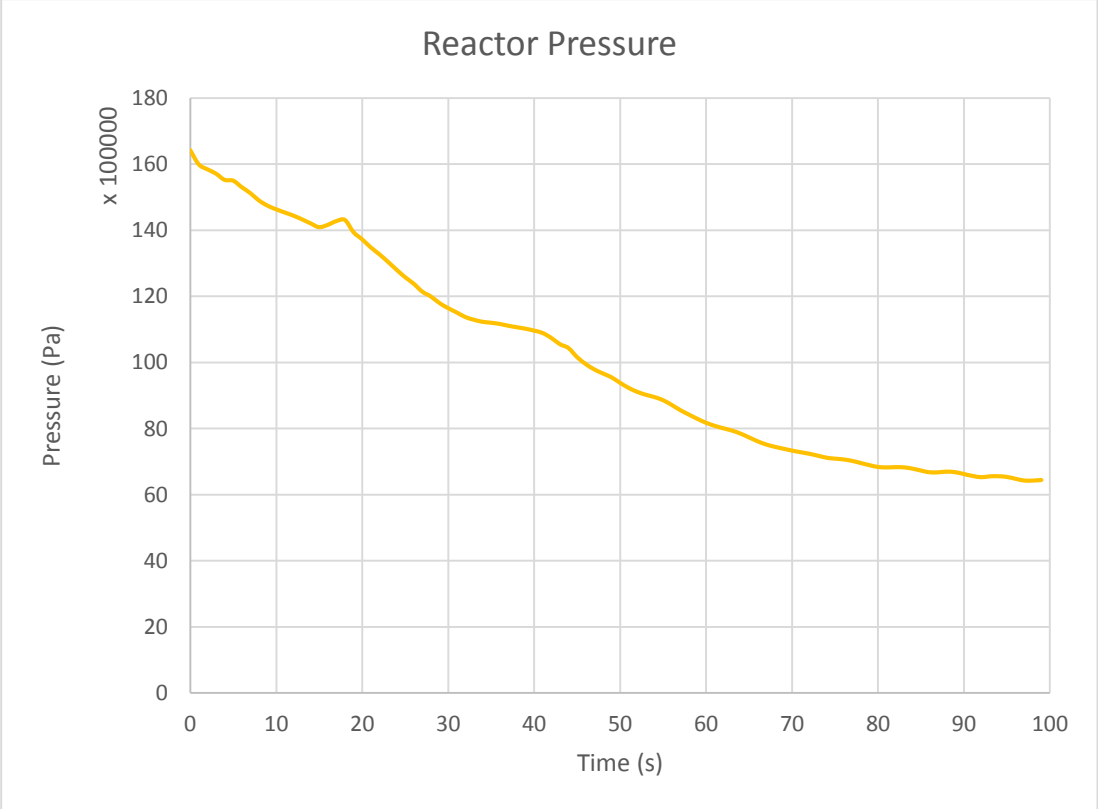

Figure 4.4.5: Reactor pressure vs time for steam leak from pressurizer transient

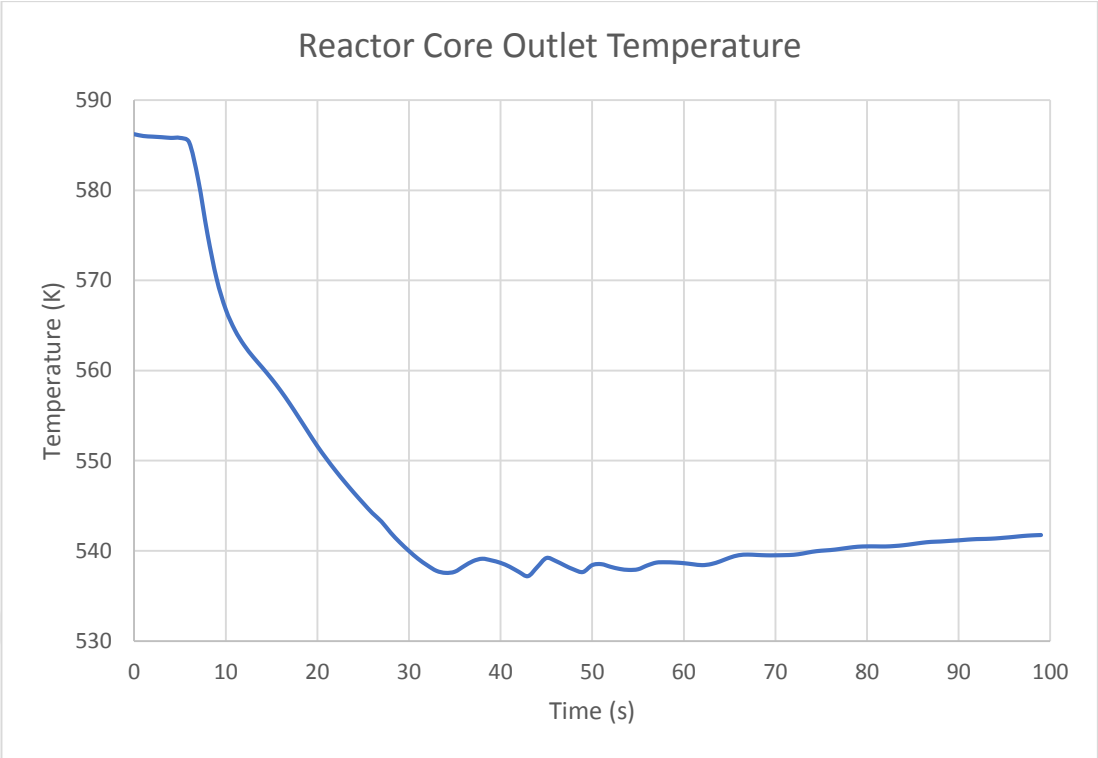

Figure 4.4.6: Reactor core outlet temperature vs time for steam leak from pressurizer transient

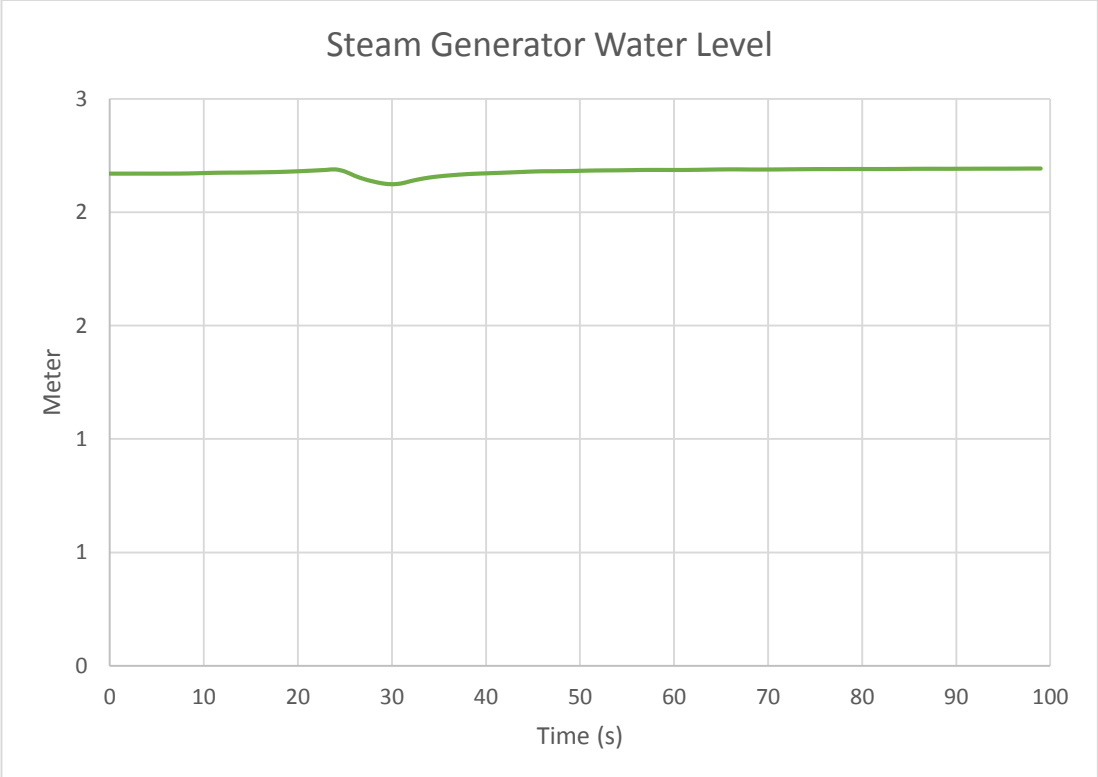

Figure 4.4.7: Steam generator water level vs time for steam leak from pressurizer transient

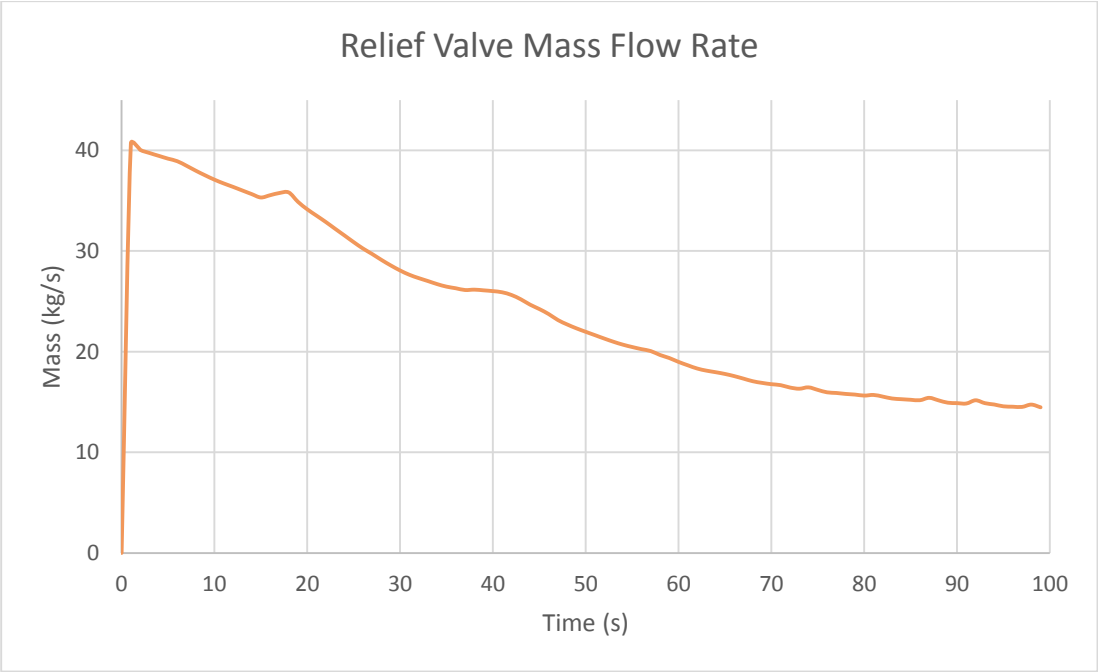

Figure 4.4.8: Steam generator water level vs time for steam leak from pressurizer transient

4.2.4 Loss of flow transient

The pump trip is defined as the loss of flow transient since during which the nominal coolant flow reduces in a way that impacts the cooling of the reactor core [69]. To simulate loss of flow with pump trip scenario, main circulation pumps defined in element numbers 107, 207, 307, and 407 were manipulated with time trip 419 that initiates at second 3500, causing pumps to coast down and entirely stop at second 104 after initiation. 4 sub-scenarios have been assessed, with only 1, 2, 3 and then respectively all 4 pumps coasted down.

```
*-------------------------------------------------------------------------
*                    TRIP CARDS
*-------------------------------------------------------------------------
*500   time  0   gt    null        0      5.0       l
419  time     0          gt   null    0      3500.       n
500      p   134010000    gt  null 0  5.6e+06   n  -1.0
499      p   134010000    gt   null  0  7.15e+06 n  -1.0
650   500  and  651   n    -1.0
```

```
1070000 kkpump1 pump
1070101  0.56745  0.0  3.0  0  90.0  1.73  0
1070108 106080002  0.56745  1.1  0.0  0001000
1070109 108010001  0.56745  1.1  0.0  0001000
1070200 113  15.75e06  578.0  8.0e-06
1070201 1  4488.3775  0.0  0.0
1070202 1  4488.3775  0.0  0.0
1070301 -2  -1  -3  -1  0  0  0
1070302  104.67  1.0  6.111111  82.5  40820.0  4.0e04  727.0
1070303  0.0  0.0  0.0  0.0  0.0
*1070310  0.0  0.0  -1.0
*pump coast down
1076100   419  time
1076101   0.0   104.67
1076102  0.5    99.337
1076103  1.0  96.59
1076104  2.0   91.92
1076105  3.0   87.52
1076106  5.0   80.05
1076107  7.0   71.94
1076108  10.0   65.659
1076109  20.0   48.00
1076110  30.0   38.265
1076111  50.0   21.739
1076112  70.0  14.45
1076113  80.0  12.629
1076114  90.0  10.826
1076115  100.0  3.776
1076116  104.0  0.0
1076117  10000.0 0.0
```

Figure 4.5.1: Trip card input for loss of flow transient

Graphical representations for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor, steam generator water level and coolant pump mass flow rate for LossofFlow1234 sub-scenario can be found in Figure 4.8 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. Both core inlet mass flow rate and coolant pump mass flow rate gradually decrease, with respect to coast down values of pumps, hence simulating the pump trip, while outlet reactor temperature shows an expected fall due to SCRAM.

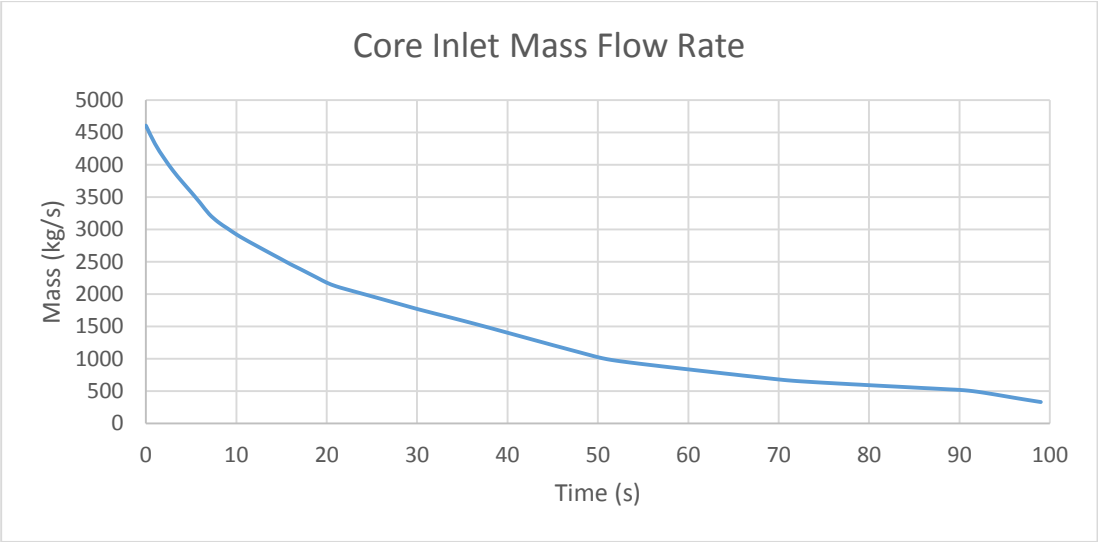

Figure 4.5.2: Core inlet mass flow rate vs time for loss of flow transient

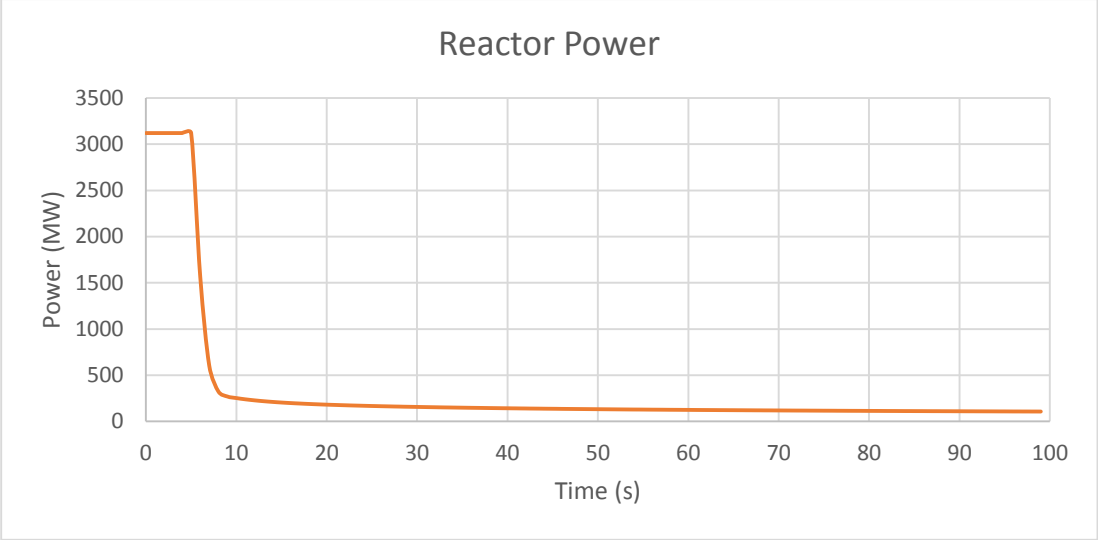

Figure 4.5.3: Reactor power vs time for loss of flow transient

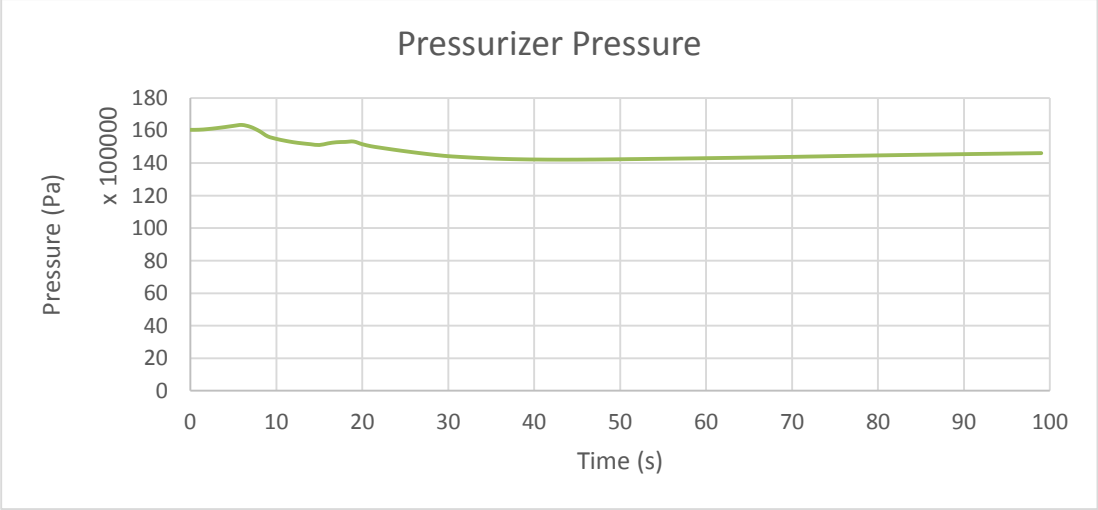

Figure 4.5.4: Pressurizer pressure vs time for loss of flow transient

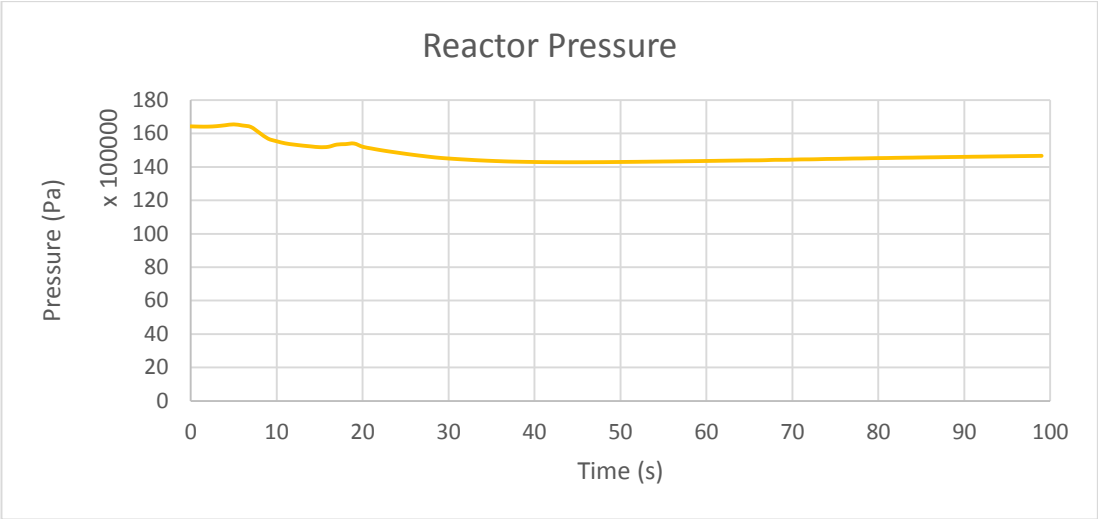

Figure 4.5.5: Reactor pressure vs time for loss of flow transient

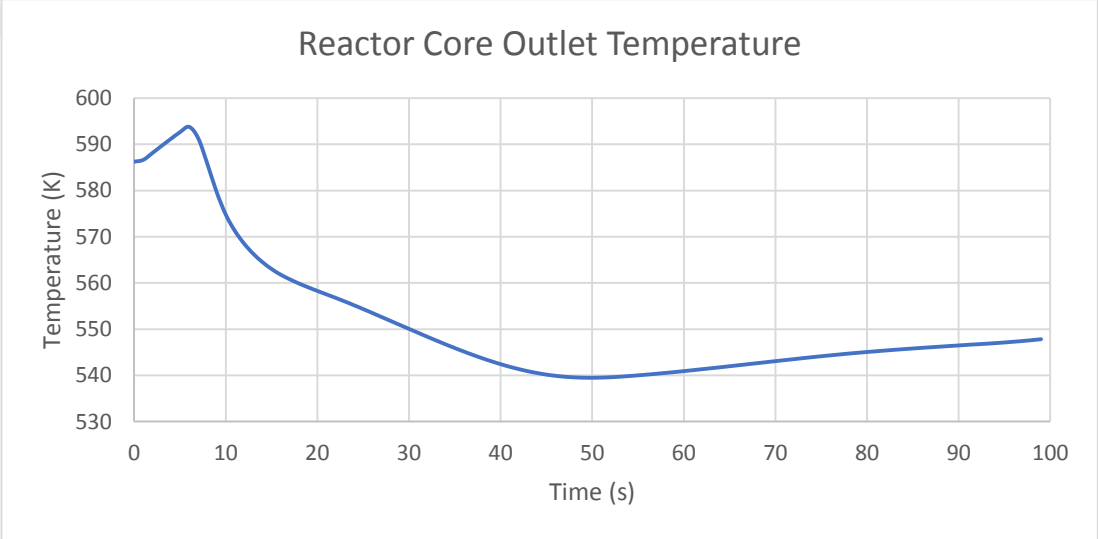

Figure 4.5.6: Reactor core outlet temperature vs time for loss of flow transient

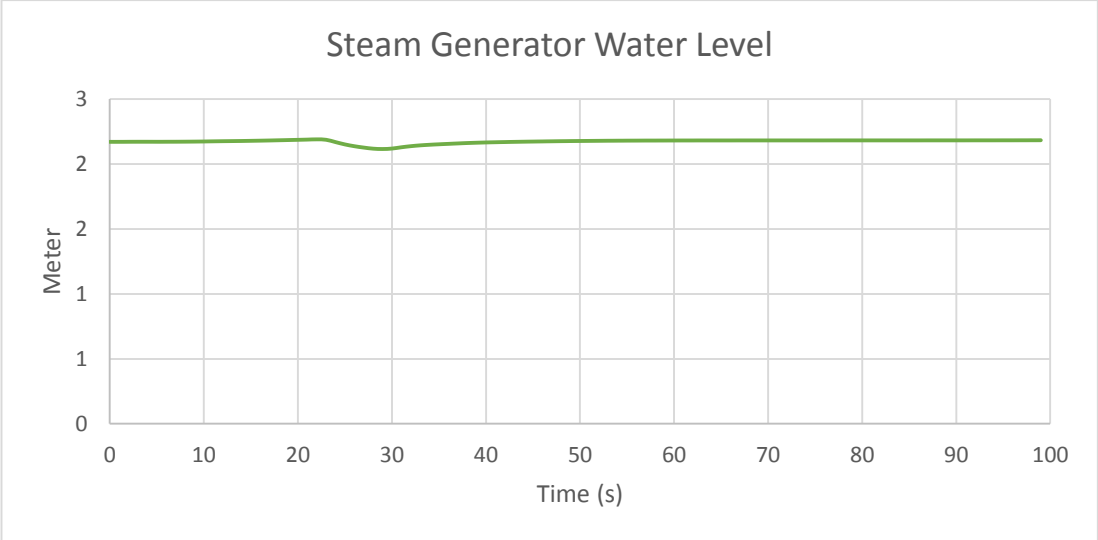

Figure 4.5.7: Steam generator water level vs time for loss of flow transient

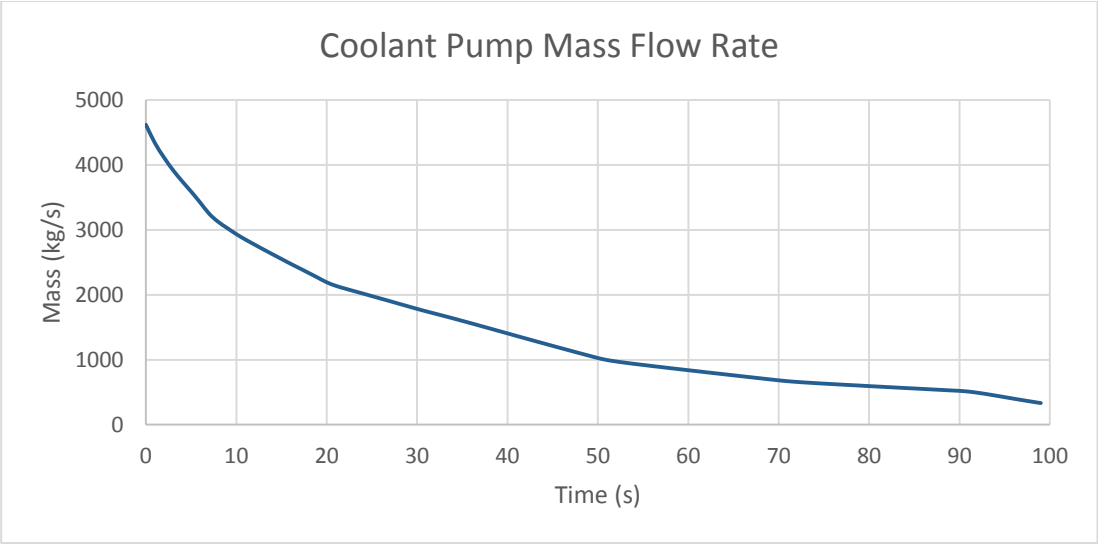

Figure 4.5.7: Coolant pump mass flow rate vs time for loss of flow transient

4.2.5 Loca in hot leg transient

A LOCA can be defined as an event that results in the loss of coolant inventory, leading to potential core damage. When there is a break in a component of the Reactor Coolant System (RCS), whether large or small, water at high pressure and temperature escapes from the loop, and there are two different scenarios for the reactor core. For large break LOCA events, a rapid loss of coolant occurs that takes a few seconds off the RCS, resulting in low coolant level and uncontrolled core heat up. For small break LOCA events, coolant is continuously lost from the RCS, the steam generators can continue to cool the core [70, 71]. In order to simulate LOCA in Hot Leg, a valve element was placed between reactor core outlet and steam generator primary side inlet in loop 1 with time trip cards 507 that initiates at second 3500 and 508 that ends at second 25.000, exceeding time of the simulation, practically causing motor valve to stay open. The water flowing into the reactor building was simulated by draining the primary coolant to time depended volume defined as element 350. To diversify for sub-scenarios, area of the valve was manipulated with respect to percentage of the sub-scenario requirement. With 5% increase in diameter resulting in 15 sub-scenarios ranging with intensity of 5% up to 90%.

```
*                    TRIP CARDS
*-------------------------------------------------------------------------
507  time      0            gt    null     0       3500.        n
508  time      0            gt    null     0       25010.        n
500       p    134010000     gt   null 0   5.6e+06    n  -1.0
499       p    134010000     gt    null   0   7.15e+06 n  -1.0
650    500   and   651    n     -1.0
651    499    or   650    n     -1.0
498       p    234010000     gt     null 0   5.6e+06 n -1.0
497       p    234010000     gt    null   0   7.15e+06 n -1.0
3490000    coll    valve *Hot Loca Valve
3490101    100010002   350000000    0.028   0.0    0.0    1100
3490201    1   0.0   0.0   0.0
3490300    mtrvlv
3490301    507    508    0.333   0.0
```

Figure 4.6.1: Trip card input for LOCA in hot leg transient

Graphical representations for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor and steam generator water level for LOCAHotLeg50 sub-scenario can be found in Figure 4.10 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. A sharp decrease in core inlet mass flow rate can be observed as expected, simulating a LOCA Hot Leg. Pressurizer tries to compensate at 4th second of mass flow rate and even though accumulators kick in 12th second resulting in a brief and small increase LOCA level is 50% and steep loss of mass flow rate follows. Pressurizer and Reactor Pressure shows a strong decrease and Reactor Temperature falls in the order of 200 K.

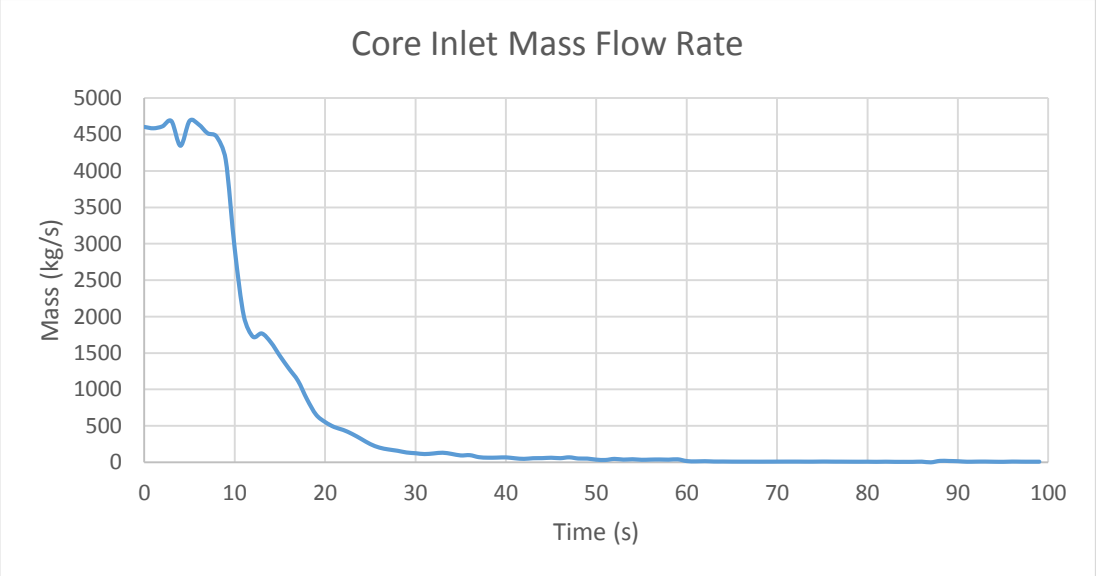

Figure 4.6.2: Core inlet mass flow rate vs time for LOCA in hot leg transient

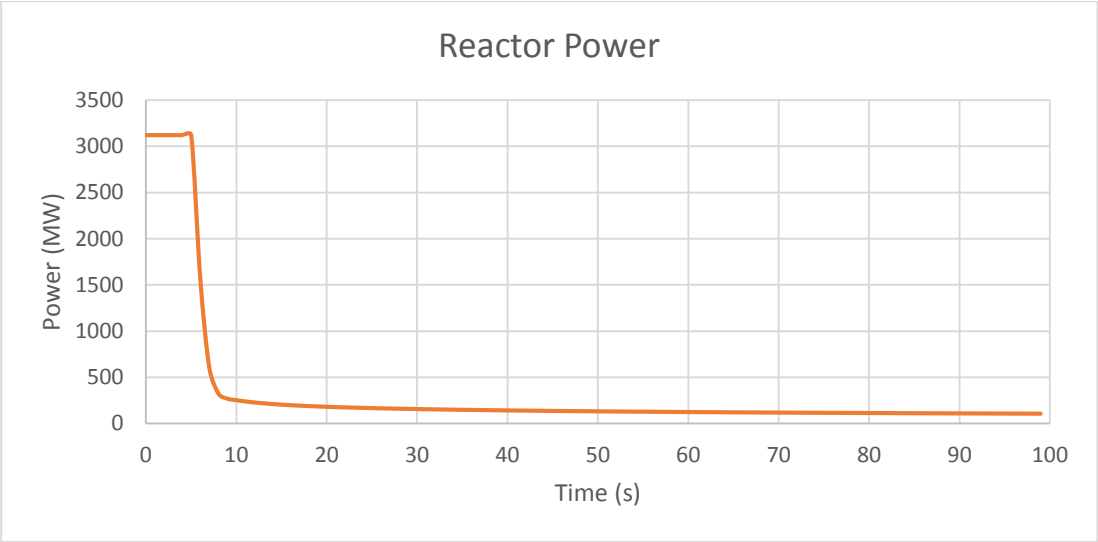

Figure 4.6.3: Reactor power vs time for LOCA in hot leg transient

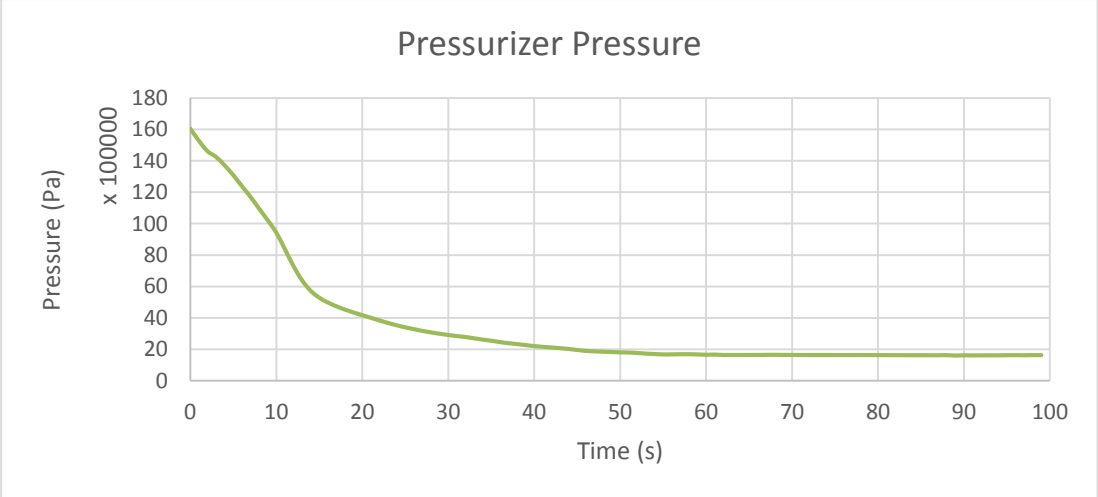

Figure 4.6.4: Pressurizer pressure vs time for LOCA in hot leg transient

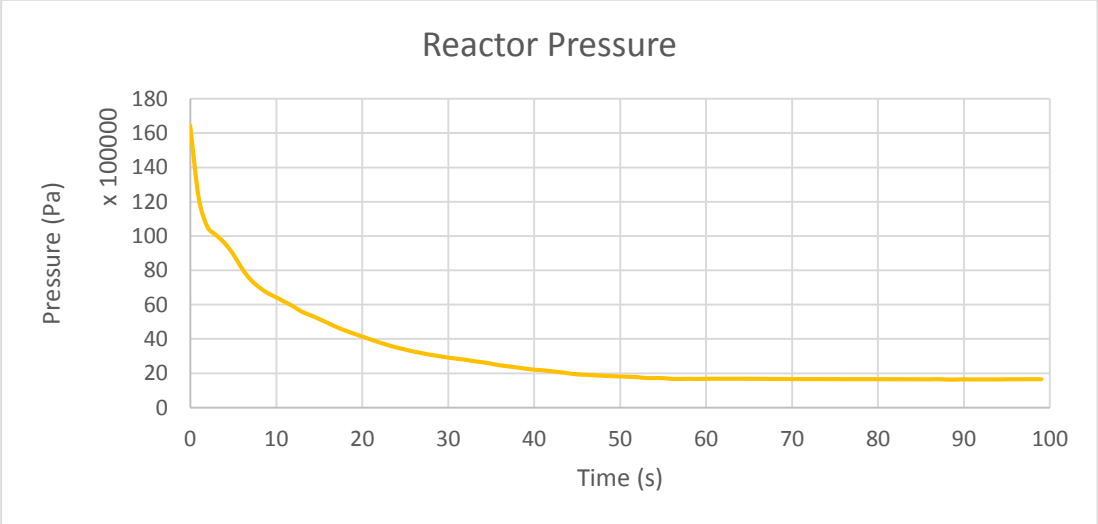

Figure 4.6.5: Reactor pressure vs time for LOCA in hot leg transient

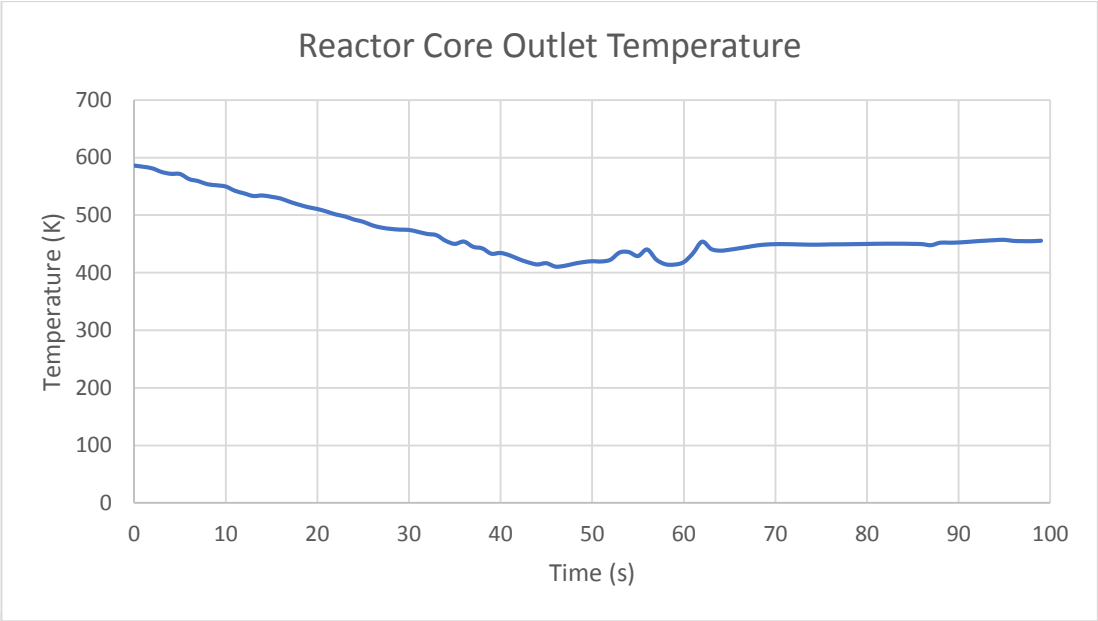

Figure 4.6.6: Reactor core outlet temperature vs time for LOCA in hot leg transient

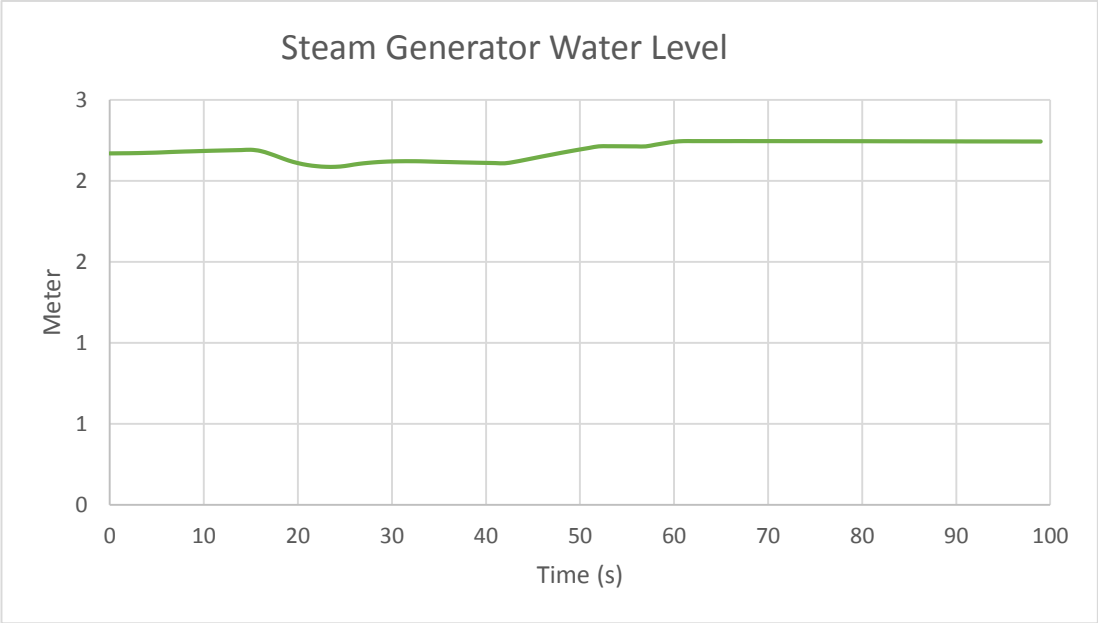

Figure 4.6.7: Steam generator water level vs time for LOCA in hot leg transient

4.2.6 Loca in cold leg

LOCA in the cold leg was simulated similar to LOCA in the hot leg only with change of the position of the valve 349. Instead of it being connected to exit of the reactor core represented with element 1000100002, it is now connected to the inlet of the reactor core represented with element 408010002. The dump again was to the time dependent volume component 350.

```
*                    TRIP CARDS
*-------------------------------------------------------------------------
507   time      0            gt    null     0       3500.        n
508   time      0            gt    null     0       25010.       n
500       p    134010000     gt   null 0   5.6e+06    n   -1.0
499       p    134010000     gt    null   0   7.15e+06 n   -1.0
650    500   and   651    n     -1.0
651    499    or   650    n     -1.0
498       p    234010000     gt     null 0   5.6e+06 n -1.0
497       p    234010000     gt    null   0   7.15e+06 n -1.0
3490000    coll    valve *Cold Loca Valve
3490101    408010002  350000000    0.028   0.0    0.0    1100
3490201    1   0.0   0.0   0.0
3490300    mtrvlv
3490301    507    508    0.333   0.0
```

Figure 4.7.1: Trip Card input for LOCA in cold leg transient

Graphical representations for reactor core outlet temperature, power, core inlet mass flow rate, pressure of pressurizer, pressure of reactor and steam generator water level for LOCAColdLeg05 sub-scenario can be found in Figure 4.12 where timeline is set to 0 where transient begins, covering 100 seconds for the period between 3500 and 3600 seconds of simulation. A timid decrease in core inlet mass flow rate can be observed as expected, in line with the level of intensity of transient hence simulating a Cold LOCA. Pressurizer tries to compensate at 4th second of mass flow rate and even though accumulators kick in 28th second resulting 40 seconds supporting since LOCA level is 5% and mass flow still exists in also there is a medium decrease. Pressurizer and reactor pressure shows an expected decrease and reactor temperature falls in the order of 60 K.

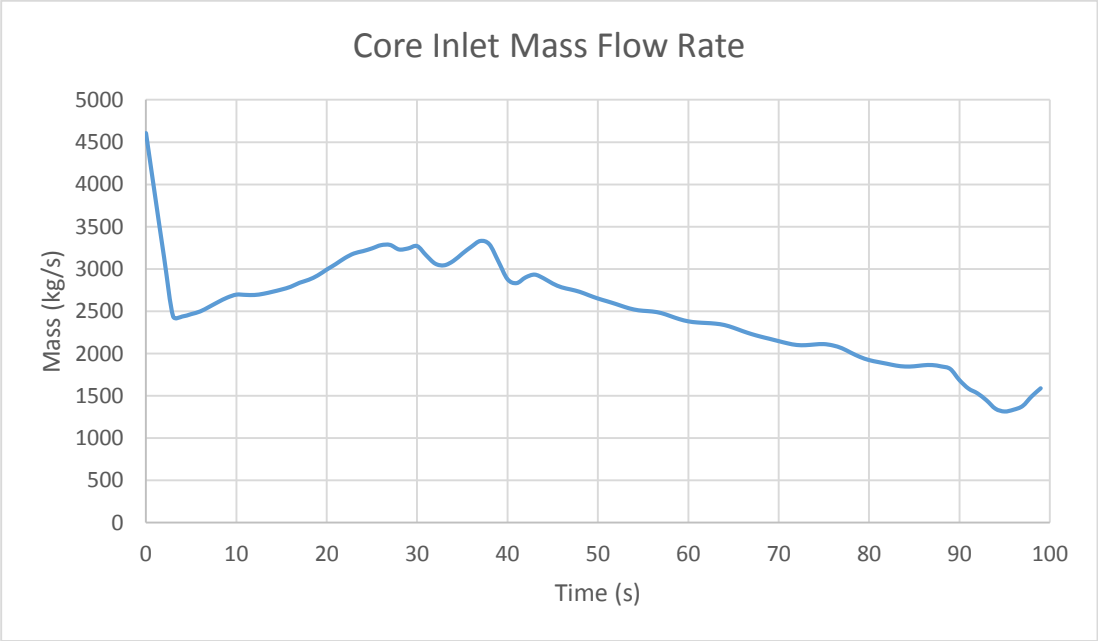

Figure 4.7.2: Core inlet mass flow rate vs time for LOCA in cold leg transient

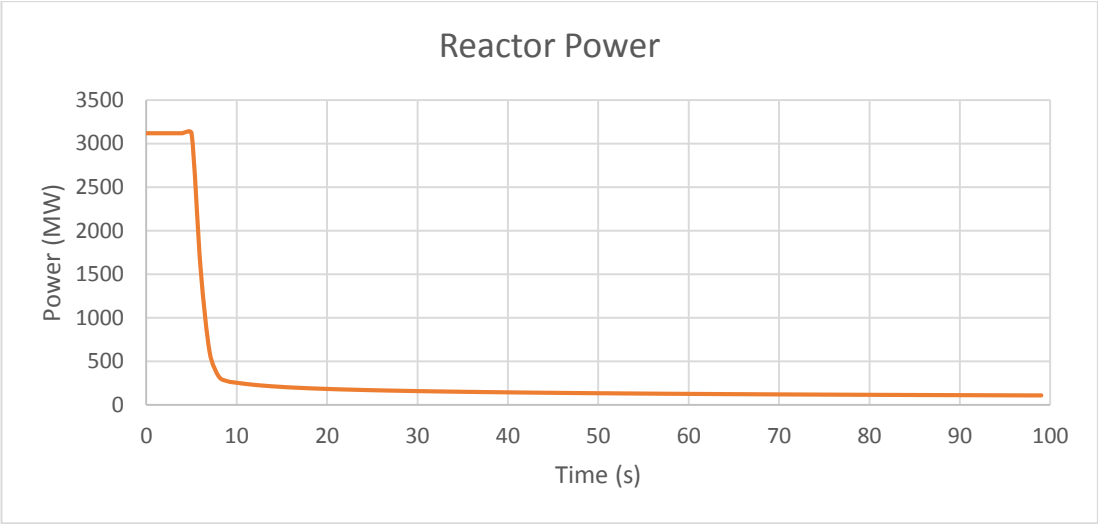

Figure 4.7.2: Reactor power vs time for LOCA in cold leg transient

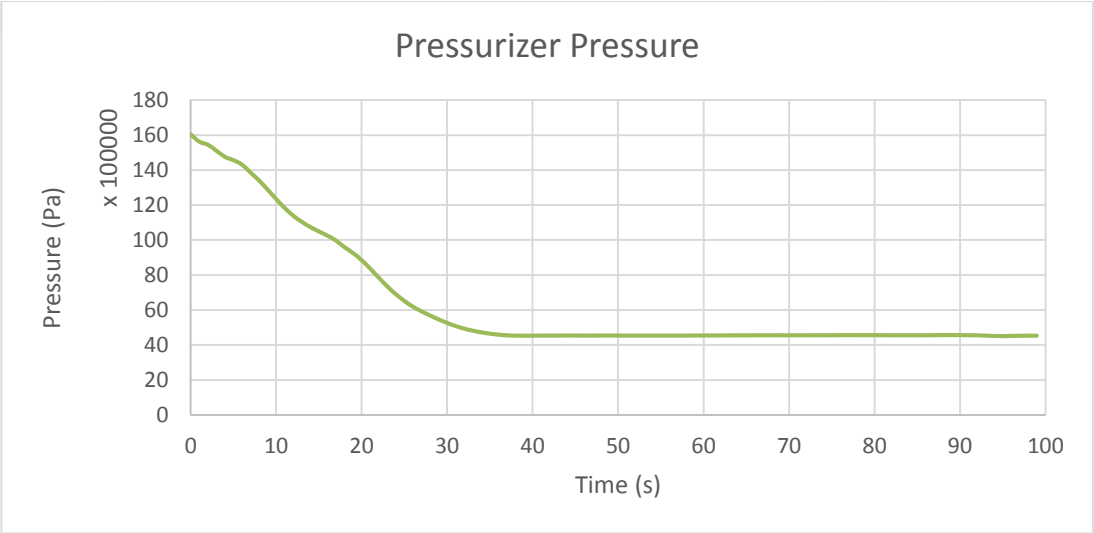

Figure 4.7.3: Pressurizer pressure vs time for LOCA in cold leg transient

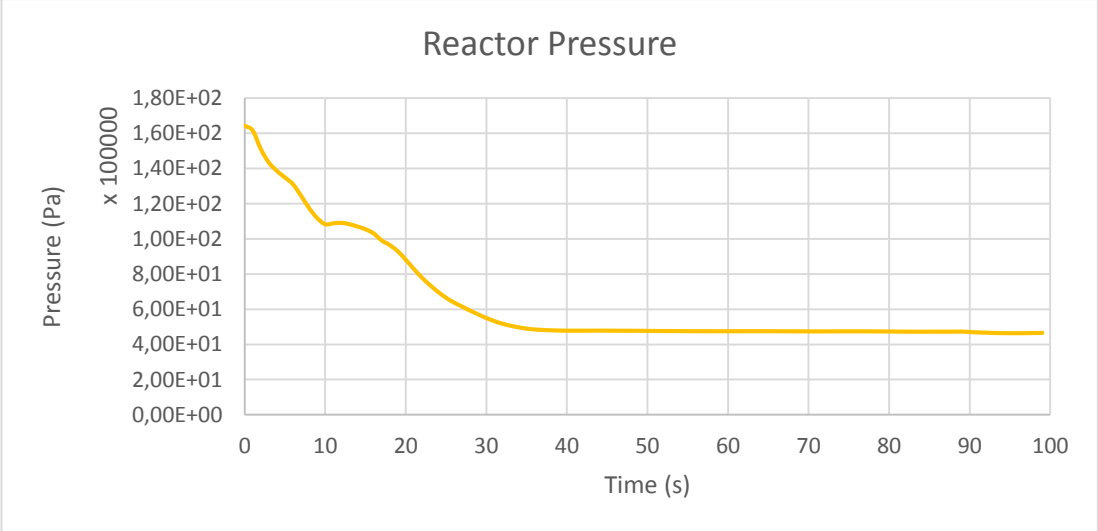

Figure 4.7.4: Reactor pressure vs time for LOCA in cold leg transient

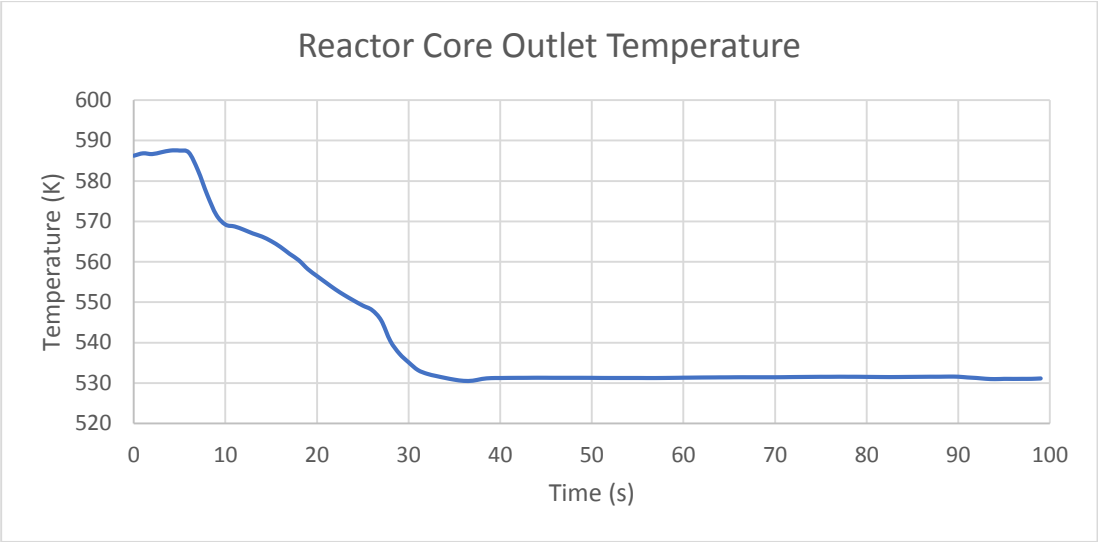

Figure 4.7.5: Reactor core outlet temperature vs time for LOCA in cold leg transient

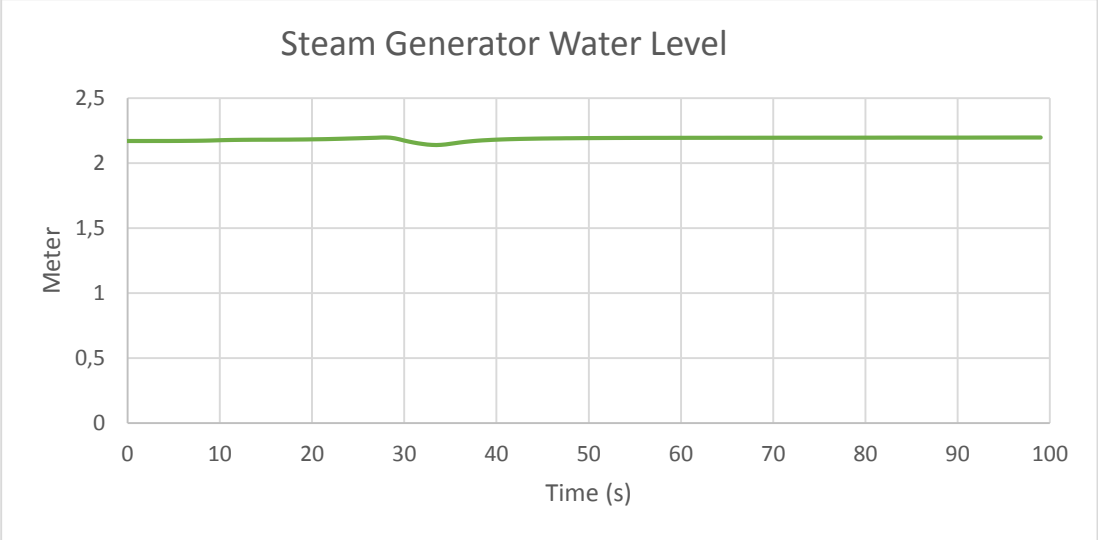

Figure 4.7.6: Steam generator water level vs time for LOCA in cold leg transient

4.3 Data Collection

Since there were many simulations to perform, a batch file was produced to parse all input files automatically and continuously. The batch file takes parameters of input file names and runs ASYST code then deletes ".r" files before starting the other simulation since ".r" file is a very large dataset and produce problems from a memory management standpoint. The batch file generated is given in Figure 4.13

```
asyst.exe -i LocaColdLeg05.i -o LocaColdLeg05.o -r LocaColdLeg05.r
del LocaColdLeg05.r
asyst.exe -i LocaColdLeg10.i -o LocaColdLeg10.o -r LocaColdLeg10.r
del LocaColdLeg10.r
asyst.exe -i LocaColdLeg15.i -o LocaColdLeg15.o -r LocaColdLeg15.r
del LocaColdLeg15.r
asyst.exe -i LocaColdLeg20.i -o LocaColdLeg20.o -r LocaColdLeg20.r
del LocaColdLeg20.r
asyst.exe -i LocaColdLeg25.i -o LocaColdLeg25.o -r LocaColdLeg25.r
del LocaColdLeg25.r
asyst.exe -i LocaColdLeg30.i -o LocaColdLeg30.o -r LocaColdLeg30.r
del LocaColdLeg30.r
asyst.exe -i LocaColdLeg35.i -o LocaColdLeg35.o -r LocaColdLeg35.r
del LocaColdLeg35.r
asyst.exe -i LocaColdLeg40.i -o LocaColdLeg40.o -r LocaColdLeg40.r
del LocaColdLeg40.r
asyst.exe -i LocaColdLeg45.i -o LocaColdLeg45.o -r LocaColdLeg45.r
del LocaColdLeg45.r
asyst.exe -i LocaColdLeg50.i -o LocaColdLeg50.o -r LocaColdLeg50.r
del LocaColdLeg50.r
asyst.exe -i LocaColdLeg55.i -o LocaColdLeg55.o -r LocaColdLeg55.r
del LocaColdLeg55.r
asyst.exe -i LocaColdLeg60.i -o LocaColdLeg60.o -r LocaColdLeg60.r
del LocaColdLeg60.r
asyst.exe -i LocaColdLeg65.i -o LocaColdLeg65.o -r LocaColdLeg65.r
del LocaColdLeg65.r
asyst.exe -i LocaColdLeg70.i -o LocaColdLeg70.o -r LocaColdLeg70.r
del LocaColdLeg70.r
asyst.exe -i LocaColdLeg75.i -o LocaColdLeg75.o -r LocaColdLeg75.r
del LocaColdLeg75.r
asyst.exe -i LocaColdLeg80.i -o LocaColdLeg80.o -r LocaColdLeg80.r
```

Figure 4.8: Batch block for ASYST simulation

All input files, after a simulation of approximately half an hour, produced an output file, consisting all simulations data. In order to obtain the data that is going to be used for AI training, 91 distinct “minor edit” cards were utilized, which outputs element and associated parameter value per second. Example of declaration in input file was supplied in figure 4.14.

```
*---------------------------------------------------------------------------
*                    MINOR EDIT VARIABLES
*---------------------------------------------------------------------------
* PRESSURE
*Pipe Minor Edit
301 mflowj 300010000
302 mflowj 301010000
303 mflowj 302010000
304 mflowj 303010000
305 mflowj 304010000
306 mflowj 305010000
307 mflowj 306010000
308 mflowj 308010000
309 mflowj 400010000
310 mflowj 401010000
311 mflowj 402010000
312 mflowj 403010000
313 mflowj 404010000
314 mflowj 405010000
315 mflowj 406010000
316 mflowj 408010000
317 mflowj 305010000
318 mflowj 508010000
319 p 300010000
320 p 301010000
321 p 302010000
322 p 303010000
323 p 304010000
324 p 305010000
```

Figure 4.9: Minor edit variables for input file

The minor edit results in data segments as seen in Figure 4.15 as block of time values for 50 seconds and repeating for respective minor edit columns. Then it is divided by unrelated data segment until it starts for the next block of 50 seconds and related minor edit data.

1 time (sec)	mflowj 307010000 (kg/sec)	mflowj 407010000 (kg/sec)	p 107010000 (Pa)	p 207010000 (Pa)	p 307010000 (Pa)	p 407010000 (Pa)	tempf 107010000 (K)	tempf 207010000 (K)	tempf 307010000 (K)
0.000000	4488.4	4488.4	1.57500E+07	1.57500E+07	1.57500E+07	1.57500E+07	578.01	578.01	578.01
1.000000	4438.6	4377.1	1.46188E+07	1.46206E+07	1.46208E+07	1.46209E+07	571.38	577.03	571.37
2.000000	4468.2	4423.2	1.50482E+07	1.50490E+07	1.50501E+07	1.50503E+07	570.33	573.13	570.32
3.000000	4466.1	4454.6	1.56677E+07	1.56678E+07	1.56672E+07	1.56698E+07	569.40	570.03	569.42
4.000000	4465.6	4468.6	1.56990E+07	1.56990E+07	1.56988E+07	1.57011E+07	568.13	568.23	568.44
5.000000	4475.1	4488.2	1.57252E+07	1.57252E+07	1.57255E+07	1.57273E+07	566.49	566.50	567.61
6.000000	4486.6	4507.9	1.57329E+07	1.57329E+07	1.57336E+07	1.57350E+07	564.70	564.70	566.68
7.000000	4495.9	4516.3	1.57214E+07	1.57214E+07	1.57223E+07	1.57236E+07	563.55	563.55	565.75
8.000000	4501.2	4513.2	1.57247E+07	1.57247E+07	1.57253E+07	1.57268E+07	563.33	563.33	565.09
9.000000	4502.2	4505.0	1.57328E+07	1.57328E+07	1.57333E+07	1.57349E+07	563.65	563.65	564.78
10.00000	4501.1	4498.1	1.57491E+07	1.57491E+07	1.57494E+07	1.57512E+07	564.06	564.06	564.73
11.00000	4500.2	4494.8	1.57736E+07	1.57736E+07	1.57738E+07	1.57757E+07	564.34	564.34	564.77
12.00000	4500.7	4494.8	1.58004E+07	1.58004E+07	1.58005E+07	1.58025E+07	564.44	564.44	564.78
13.00000	4502.8	4496.8	1.58252E+07	1.58252E+07	1.58253E+07	1.58273E+07	564.38	564.39	564.69
14.00000	4506.3	4499.9	1.58484E+07	1.58485E+07	1.58485E+07	1.58506E+07	564.22	564.22	564.51
15.00000	4510.4	4503.2	1.58657E+07	1.58657E+07	1.58658E+07	1.58678E+07	564.00	564.00	564.25
16.00000	4514.1	4505.8	1.58744E+07	1.58744E+07	1.58745E+07	1.58765E+07	563.78	563.78	563.99
17.00000	4517.0	4507.7	1.58785E+07	1.58785E+07	1.58786E+07	1.58806E+07	563.59	563.59	563.75
18.00000	4518.9	4508.7	1.58799E+07	1.58799E+07	1.58800E+07	1.58820E+07	563.46	563.46	563.57
19.00000	4519.9	4509.2	1.58808E+07	1.58808E+07	1.58809E+07	1.58829E+07	563.37	563.38	563.45
20.00000	4520.5	4509.6	1.58829E+07	1.58829E+07	1.58830E+07	1.58851E+07	563.31	563.31	563.37
21.00000	4521.1	4510.3	1.58861E+07	1.58861E+07	1.58862E+07	1.58882E+07	563.25	563.25	563.30
22.00000	4521.8	4511.2	1.58900E+07	1.58900E+07	1.58901E+07	1.58922E+07	563.17	563.17	563.24
23.00000	4522.9	4512.4	1.58947E+07	1.58947E+07	1.58948E+07	1.58968E+07	563.08	563.08	563.16
24.00000	4524.1	4513.8	1.58982E+07	1.58982E+07	1.58983E+07	1.59003E+07	562.97	562.97	563.06
25.00000	4525.6	4515.3	1.59015E+07	1.59015E+07	1.59016E+07	1.59036E+07	562.85	562.85	562.94
26.00000	4527.1	4516.8	1.59041E+07	1.59041E+07	1.59041E+07	1.59062E+07	562.73	562.73	562.82

Figure 4.10: Minor edit data output

For an AI training, data must be in format of continuous rows for minor edit columns. After pasting the related data to excel, two macro codes were utilized to manipulate the data to achieve 100 seconds of transient simulation for each sub-scenario. The visual studio code block that marks related rows and converts repeating row patterns to columns are given in Figure 4.16 and Figure 4.17, respectively.

```
"Sub FindValue()

    Dim i As Long
    Dim j As Long
    Dim k As String
    Dim l As String
    Dim c As Range
    Dim d As Range
    Dim firstAddress As String

    With Worksheets(1).Range(""A1:A400000"")
        Set c = .Find(1, LookIn:=xlValues, lookat:=xlWhole)
         If Not c Is Nothing Then
         Do
       i = CDbl(Right(c.Address, Len(c.Address) - 3))
       j = i + 53
       k = ""A"" & CStr(i) & "":A"" & CStr(j)
      Worksheets(""Sheet1"").Range(k).Value = 3
       Set c = .FindNext(c)
       Loop While Not c Is Nothing
        End If

   End With
End Sub
```

Figure 4.11: Macro code for converting rows to columns

```
"Sub Transpose_my_cells()
    Dim i As Integer
    Dim j As Integer
    Dim z As Integer
    Dim x As Integer
    Dim Y As Integer
    Dim k As String
    Dim l As String
    Dim range1 As String
    Dim range2 As String
    Dim c As Range
    Dim d As Range
    Dim firstAddress As String
    k = ""A""
    l = "":K""
    i = 1
    j = 54
    x = 12
    For z = 1 To 22
    i = i + 54
    j = j + 54
    range1 = k & CStr(i) & l & CStr(j)
    Worksheets(""Sheet2"").Range(range1).Copy
    Worksheets(""Sheet2"").Cells(1, x).PasteSpecial
    x = x + 11

    Next z
End Sub"
```

Figure 4.12: Macro code for row marking

The resulting structured data segment, that is executable and usable for AI implementation is seen in Figure 4.18.

TIME	flowj3000100	flowj3010100	flowj3020100	cntrlvar1	cntrlvar2	cntrlvar3	mpf5050100	flowj1510000	flowj1830000	flowj5290000	flowj5300000	TRANSIENT NO	TRASIENT	FLAG	TRANSIENT GENEL	IENT GENEL
3500	4616.4	3130.2	1486.2	2.1701	2.1701	2.1701	555.1	4616.4	417.27	0	0	1	STEADYSTATE	TRAINING	SS	1
3501	4616.4	3130.2	1486.2	2.1699	2.1701	2.17	555.1	4616.4	415.51	0	0	1	STEADYSTATE	TRAINING	SS	1
3502	4616.4	3130.2	1486.2	2.1699	2.17	2.1699	555.1	4616.4	415.52	0	0	1	STEADYSTATE	TRAINING	SS	1
3503	4616.4	3130.2	1486.2	2.17	2.17	2.17	555.1	4616.4	417.04	0	0	1	STEADYSTATE	TRAINING	SS	1
3504	4616.4	3130.2	1486.2	2.1701	2.17	2.1701	555.1	4616.4	417.23	0	0	1	STEADYSTATE	TRAINING	SS	1
3505	4616.4	3130.2	1486.2	2.17	2.1699	2.1701	555.1	4616.4	416.79	0	0	1	STEADYSTATE	TRAINING	SS	1
3506	4616.4	3130.2	1486.2	2.1701	2.1699	2.1701	555.1	4616.4	417.41	0	0	1	STEADYSTATE	TRAINING	SS	1
3507	4616.4	3130.2	1486.2	2.1699	2.17	2.1699	555.1	4616.4	415.47	0	0	1	STEADYSTATE	TRAINING	SS	1
3508	4616.4	3130.2	1486.2	2.1699	2.1701	2.1699	555.1	4616.4	415.43	0	0	1	STEADYSTATE	TRAINING	SS	1
3509	4616.4	3130.2	1486.2	2.17	2.17	2.1699	555.1	4616.4	417.01	0	0	1	STEADYSTATE	TRAINING	SS	1
3510	4616.4	3130.2	1486.2	2.1701	2.1699	2.17	555.1	4616.4	417.21	0	0	1	STEADYSTATE	TRAINING	SS	1
3511	4616.4	3130.2	1486.2	2.17	2.17	2.1701	555.1	4616.4	416.77	0	0	1	STEADYSTATE	TRAINING	SS	1
3512	4616.4	3130.2	1486.2	2.1701	2.17	2.1701	555.1	4616.4	417.31	0	0	1	STEADYSTATE	TRAINING	SS	1
3513	4616.4	3130.2	1486.2	2.1699	2.1701	2.17	555.1	4616.4	415.33	0	0	1	STEADYSTATE	TRAINING	SS	1
3514	4616.4	3130.2	1486.2	2.1699	2.1699	2.1699	555.1	4616.4	415.38	0	0	1	STEADYSTATE	TRAINING	SS	1
3515	4616.4	3130.2	1486.2	2.17	2.1699	2.1699	555.1	4616.4	416.39	0	0	1	STEADYSTATE	TRAINING	SS	1
3516	4616.4	3130.2	1486.2	2.1701	2.17	2.17	555.1	4616.4	417.2	0	0	1	STEADYSTATE	TRAINING	SS	1
3517	4616.4	3130.2	1486.2	2.17	2.1701	2.17	555.1	4616.4	416.28	0	0	1	STEADYSTATE	TRAINING	SS	1
3518	4616.4	3130.2	1486.2	2.1699	2.17	2.1701	555.1	4616.4	415.19	0	0	1	STEADYSTATE	TRAINING	SS	1
3519	4616.4	3130.2	1486.2	2.1699	2.1699	2.1701	555.1	4616.4	415.91	0	0	1	STEADYSTATE	TRAINING	SS	1
3520	4616.4	3130.2	1486.2	2.17	2.17	2.1699	555.1	4616.4	416.97	0	0	1	STEADYSTATE	TRAINING	SS	1
3521	4616.4	3130.2	1486.2	2.1701	2.17	2.1699	555.1	4616.4	417.21	0	0	1	STEADYSTATE	TRAINING	SS	1
3522	4616.4	3130.2	1486.2	2.17	2.1701	2.1699	555.1	4616.4	416.7	0	0	1	STEADYSTATE	TRAINING	SS	1
3523	4616.4	3130.2	1486.2	2.17	2.1699	2.17	555.1	4616.4	416.5	0	0	1	STEADYSTATE	TRAINING	SS	1
3524	4616.4	3130.2	1486.2	2.1699	2.1699	2.1701	555.1	4616.4	415.25	0	0	1	STEADYSTATE	TRAINING	SS	1
3525	4616.4	3130.2	1486.2	2.1699	2.17	2.1701	555.1	4616.4	415.94	0	0	1	STEADYSTATE	TRAINING	SS	1
3526	4616.4	3130.2	1486.2	2.17	2.1701	2.17	555.1	4616.4	416.98	0	0	1	STEADYSTATE	TRAINING	SS	1
3527	4616.4	3130.2	1486.2	2.1701	2.17	2.1699	555.1	4616.4	417.19	0	0	1	STEADYSTATE	TRAINING	SS	1
3528	4616.4	3130.2	1486.2	2.1699	2.1699	2.1699	555.1	4616.4	415.5	0	0	1	STEADYSTATE	TRAINING	SS	1
3529	4616.4	3130.2	1486.2	2.1699	2.1699	2.17	555.1	4616.4	415.26	0	0	1	STEADYSTATE	TRAINING	SS	1
3530	4616.4	3130.2	1486.2	2.17	2.17	2.1701	555.1	4616.4	416.79	0	0	1	STEADYSTATE	TRAINING	SS	1
3531	4616.4	3130.2	1486.2	2.1701	2.1701	2.1701	555.1	4616.4	417.15	0	0	1	STEADYSTATE	TRAINING	SS	1
3532	4616.4	3130.2	1486.2	2.1701	2.17	2.1701	555.1	4616.4	417.07	0	0	1	STEADYSTATE	TRAINING	SS	1
3533	4616.4	3130.2	1486.2	2.1701	2.1699	2.1699	555.1	4616.4	417.12	0	0	1	STEADYSTATE	TRAINING	SS	1
3534	4616.4	3130.2	1486.2	2.1699	2.17	2.1699	555.1	4616.4	415.23	0	0	1	STEADYSTATE	TRAINING	SS	1
3535	4616.4	3130.2	1486.2	2.1699	2.17	2.1699	555.1	4616.4	415.13	0	0	1	STEADYSTATE	TRAINING	SS	1
3536	4616.4	3130.2	1486.2	2.17	2.1701	2.1699	555.1	4616.4	416.57	0	0	1	STEADYSTATE	TRAINING	SS	1
3537	4616.4	3130.2	1486.2	2.17	2.1699	2.17	555.1	4616.4	417.07	0	0	1	STEADYSTATE	TRAINING	SS	1
3538	4616.4	3130.2	1486.2	2.1701	2.1699	2.1701	555.1	4616.4	417.26	0	0	1	STEADYSTATE	TRAINING	SS	1
3539	4616.4	3130.2	1486.2	2.1701	2.17	2.1701	555.1	4616.4	417.43	0	0	1	STEADYSTATE	TRAINING	SS	1
3540	4616.4	3130.2	1486.2	2.1699	2.1701	2.1701	555.1	4616.4	415.33	0	0	1	STEADYSTATE	TRAINING	SS	1
3541	4616.4	3130.2	1486.2	2.1699	2.1701	2.1701	555.1	4616.4	415.08	0	0	1	STEADYSTATE	TRAINING	SS	1
3542	4616.4	3130.2	1486.2	2.17	2.17	2.1699	555.1	4616.4	416.27	0	0	1	STEADYSTATE	TRAINING	SS	1
3543	4616.4	3130.2	1486.2	2.17	2.1699	2.1699	555.1	4616.4	417.01	0	0	1	STEADYSTATE	TRAINING	SS	1
3544	4616.4	3130.2	1486.2	2.1701	2.1699	2.1699	555.1	4616.4	417.25	0	0	1	STEADYSTATE	TRAINING	SS	1
3545	4616.4	3130.2	1486.2	2.1701	2.17	2.17	555.1	4616.4	417.3	0	0	1	STEADYSTATE	TRAINING	SS	1
3546	4616.4	3130.2	1486.2	2.1699	2.1701	2.1701	555.1	4616.4	415.39	0	0	1	STEADYSTATE	TRAINING	SS	1
3547	4616.4	3130.2	1486.2	2.1699	2.1701	2.1701	555.1	4616.4	414.97	0	0	1	STEADYSTATE	TRAINING	SS	1
3548	4616.4	3130.2	1486.2	2.1699	2.1701	2.17	555.1	4616.4	415.81	0	0	1	STEADYSTATE	TRAINING	SS	1
3549	4616.4	3130.2	1486.2	2.17	2.1699	2.17	555.1	4616.4	416.91	0	0	1	STEADYSTATE	TRAINING	SS	1
3550	4616.4	3130.2	1486.2	2.1701	2.1699	2.1699	555.1	4616.4	417.18	0	0	1	STEADYSTATE	TRAINING	SS	1
3551	4616.4	3130.2	1486.2	2.1701	2.17	2.17	555.1	4616.4	417.27	0	0	1	STEADYSTATE	TRAINING	SS	1
3552	4616.4	3130.2	1486.2	2.1699	2.17	2.17	555.1	4616.4	415.32	0	0	1	STEADYSTATE	TRAINING	SS	1

Figure 4.13: Structured data segment

5. DEVELOPMENT OF AI MODELS

5.1 Formatting and organizing data

Training data in general consists of a set of data points utilized to instruct ML models and data mining models. It helps systems to identify useful patterns for prediction, decision-making, and related tasks. Training data is pivotal for constructing high-quality and unbiased models: it fundamentally determines the upper bounds of evaluation metrics. Moreover, without appropriate training data, the positive effect of other components, such as model architectures and hyper parameters, could be diminished [72]. Outliers are examples that differ considerably from other examples in the same target space. Outliers can, for example, be large errors between predicted and true target values. Detecting outliers is important in safety-critical applications, and interesting interactions may be missed in the presence of extreme examples. There are two basic scenarios of outlier detection: in the first one, it is known a priori whether or not the data set is clean therefore the aim is to process data set in such a way to lessen the impact of the most effectual outliers; in the second one, the data is not examined therefore the aim is to provide a rule that assigns a binary label to examples [73].

5.2 Setting up Training and Validation Sets

The training and validation data of this work has been produced through simulations of coded software hence removing outliers is not a necessary task. The training dataset therefore was used to set the AI model parameters. The validation dataset was utilized during the training process to assess the AI model's ability to generalize a dataset that is distinct from the training set. Validation dataset is crucial for evaluating the performance of the AI model since it makes it possible to verify the robustness of the model, in addition to its overall performance [74]. It is important that validation set must present well-distributed data in all classes of the label.

One problem with random sampling arises when dealing with datasets composed of different classes. For a multi-class classification task, randomly selected data points

may not guarantee that each class is adequately represented in the validation set [75]. As a result, it may happen that either a class is not represented in the validation set at all or that one class is significantly under-represented compared to other classes. In general, 80% training and 20% validation dataset distribution is standard in AI and ML practices in literature. Therefore in the data acquisition process, 91 features were normalized for training data set compromised of 4085 rows using their respective minimum and maximum values to train the methods, while 1030 rows were reserved for validating their performance. In this study, every sub-scenario typically has 100 seconds of simulation data except LocaColdLeg80 and LocaColdLeg85 which have simulation time of 43 seconds and 29 seconds respectively due to simulation constraints. However, 80% to 20% training to validation ratio was kept constant across all sub-scenarios.

In this study, in order to present under-representation, stratified sampling was used, where the data set was first divided into strata (subsets) so that all data points within the stratified subset belonged to the same class. Afterwards, random sampling was applied on these strata to form the training and validation sets. Stratified sampling is crucial in the case of imbalanced classes, where the number of examples in one class is orders of magnitude smaller than in other classes [76]. To compensate for lack of representation in the cases, every sub-scenario was presented equally in training being 80% and validation being 20% with cases represented both before and after SCRAM initiation hence covering both cases of the simulations.

Data normalization serves as a fundamental pre-processing technique that improves the accuracy of forecasting and prediction models. This process entails converting the existing data range into a new, standardized range. Normalization plays a vital role in ensuring consistency across different prediction and forecasting methods. By standardizing the range of independent variables or features within a dataset, data normalization boosts the reliability and comparability of various predictive models, leading to more stable results [77]. Normalization is the procedure of modifying numerical columns to conform to a standardized scale, which is crucial for datasets that include attributes with different units or magnitudes. The main goal of normalization is to create a uniform scale for the data while maintaining the natural differences in value ranges. This process typically includes rescaling the features to a

specified range, commonly between 0 and 1, or modifying them so that they have a mean of 0 and a standard deviation of 1 [78].

In this work, min-max scaler function in Sci-Kit Library was utilized for 91 distinct features, providing a distributed data set between 0 and 1, normalizing the features with the equation 5.1.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (5.1)$$

5.2.1 Regularization

Regularization is a technique used in ML to reduce the likelihood of overfitting. Overfitting happens when a model becomes excessively focused on the training dataset, incorporating noise and outliers, which adversely affects its performance on new data. Essentially, regularization introduces a penalty for excessive complexity in the model, promoting a simpler and more generalized approach. Consequently, reducing the likelihood of making extreme predictions influenced by noise within the data as provided in figure 5.1 [79]. In the context of this dissertation however, regularization was not applied.

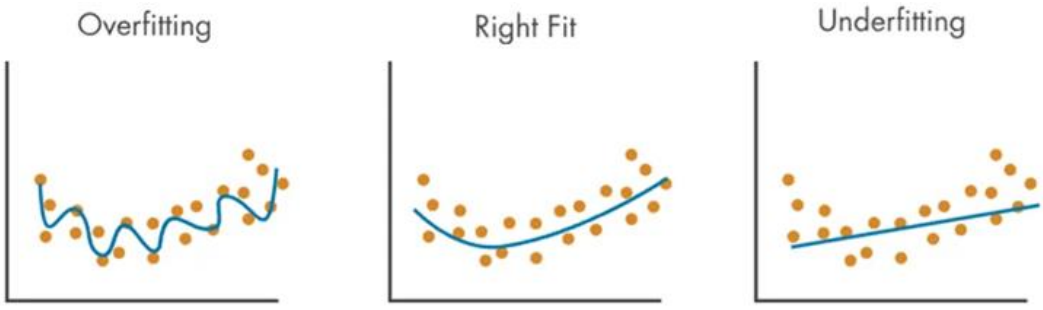

Figure 5.1: Representation of overfitting-right fit-underfitting

5.3 Python and Colab

Python is a programming language that deals with all programming paradigms that is developed by Guida van Rossum in the late 1980s and its implementation was published in February, 1991 [80]. It is the go-to programming language when one starts programming, due to its syntax making it easy to learn and understand. Python supports object-oriented programming (OOP), procedural programming, and functional programming. For data science and ML, many libraries and frameworks have been developed based on its syntax [81]. Launched by Google in mid-2017,

Colab (short for “Colaboratory”) allows Python code execution in a web browser without the need for installation. It provides storage in Google Drive and enables collaboration via shareable links. In addition, it supports popular libraries like TensorFlow, Keras, and OpenCV, facilitating data import and visualization. Researchers and developers often adopt this versatile tool, appreciating its access to Google’s cloud capacity through GPU or TPU socket nodes. Similar to Jupyter Notebook, Colab utilizes the IPython kernel and Mozilla’s JavaScript-based Jupyter Notebook viewer. Its key distinction lies in its cloud-based architecture—users run Python code in the browser rather than directly on the device’s “local” kernel. Code execution occurs on a physical server, and results are shared through the browser with no requirements for local resource installation or additional setup. With a recommended Chrome or Firefox browser, users can access Colab by logging into their Google account. Supported file types include notebooks, Python scripts, and markdown files, and users can upload files from their local machine or directly from Google Drive. All files are stored in Notebook format, and when a notebook is opened, its environment is set up automatically, replicating the programming environment from the last save. Resources, external libraries, and local files are rendered accessible within seconds [82].

To conduct a thorough analysis, various libraries must be imported in Colab. By default, certain libraries come pre-loaded. Importing commonly utilized libraries can save time later in the process. For reference, the libraries and their functionalities are as follows:

NumPy: A Python library designed to handle large, multi-dimensional arrays and matrices, providing a set of high-level mathematical functions for operations on these arrays. It includes numerous mathematical functionalities, including statistics, linear algebra, Fourier transforms, and random number generation, among others. Typically, NumPy is imported using the alias np.

Pandas: A Python library that offers efficient, versatile, and clear data structures intended for straightforward and intuitive interaction with relational or labeled data. This library is built on top of NumPy. It was designed for web scraping, much like BeautifulSoup, but can also be used for cleaning and analyzing data. Pandas is usually imported as pd.

Matplotlib: A Python library, along with its numerical mathematics extension NumPy, offers an object-oriented API designed for integrating plots into applications that utilize general-purpose GUI toolkits. It also includes various modules to assist with common plotting tasks. Additionally, it can be utilized for combining plots and images as well as for easier graphing. Matplotlib is usually imported as plt.

Seaborn: Seaborn is a Python library designed for data visualization, built on top of Matplotlib. It offers a high-level interface for creating visually appealing statistical graphics and is pre-installed on Google's servers. It is commonly imported using the abbreviation sns [83].

5.4 Codebase for trainings

Utilizing Google Colab and Scikit-learn library, KNN, Random Forest, Gradient Boosting, Decision Trees, Logistic Regression, Support Vector, Naïve Bayes, Multi-Layer Perceptron methods were implemented. Basically code block takes 80% of samples as x-training and y-training and fits the model and then forecasts with x-validation to classify for y-validation file. Following the flow chart at Figure 5.2 for every transient, the code base can be seen for models in Figure A.1. For every model, a common block of code was implemented for mapping training and validation data sets as csv files and also transforming output to csv file. Parameters for hyper tuning is intrinsically at the line of classifier declaration and optimization has taken place with manipulating these parameters. For visualization of learning curve and a training scores separate code block was implemented for each model as seen in the Figure A.2.

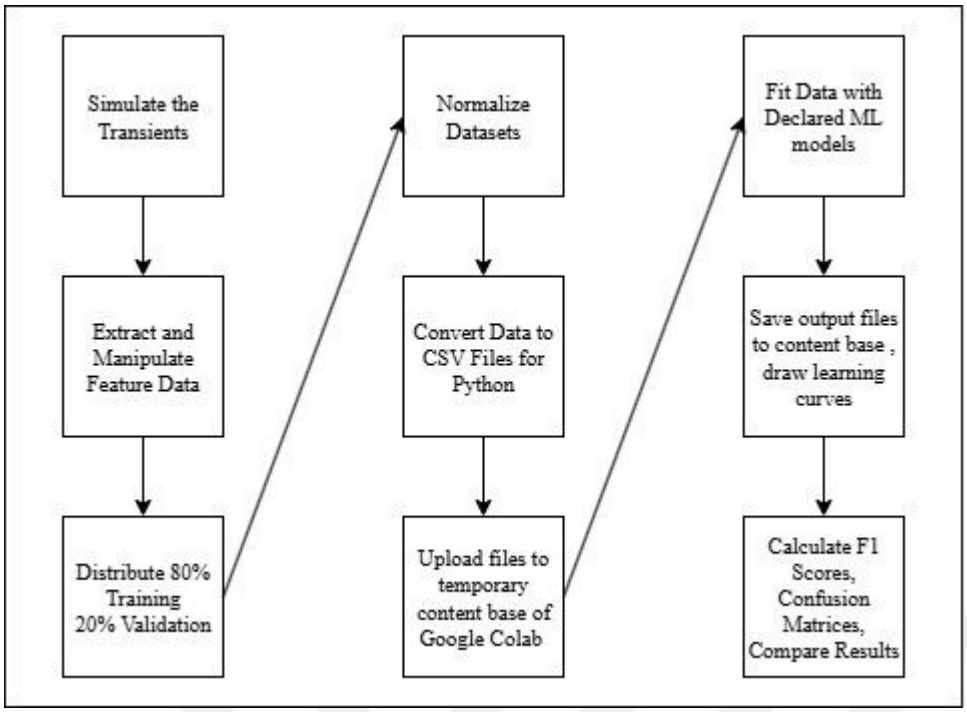

Figure 5.2: Flowchart for codebase

6. RESULTS

Four of the most widely used performance metrics for AI methods that are accuracy, precision, recall, and F1-score, confusion matrix, feature importance, and learning curve visualization were applied for models and their performance was assessed with respect to these metrics.

6.1 Results for Main Transients Identification

For validating the basic level of the training data's ability of differentiating between sub-scenarios, a model for main transient detection was implemented. A total of 1030 rows of transient data were added to the method for validation. The performance indicators of the ML methods employed are presented in Table 6.1. According to Table 6.1 the Random Forest performed slightly better than Gradient Boosting for accuracy and F1-Score, hence Random Forest was identified as the most robust option among the eight alternatives when evaluated by using four metrics in Table 6.1 with confusion matrix provided in Table 6.2.

Table 6.1: Accuracy, precision, recall and F1-score of methods used for main transient identification

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Random Forest	99.51	99.50	100.00	99.75
Gradient Boosting	99.42	99.50	100.00	99.70
Logistic Regression	97.67	97.62	100.00	98.80
K-Nearest Neighbor	97.09	97.13	99.90	98.49
Decision Trees	96.02	95.94	100.00	97.93
Support Vector	93.40	95.25	97.96	96.59
Naïve Bayes	91.36	91.19	100.00	95.39
Multi-Layer Perceptron	80.39	81.98	97.64	89.13

Table 6.2: Confusion matrix for main transient identification

	Transient	Rod Withdrawal	Steam Leak from Pressurizer	Loss of Flow	LOCA Hot Leg	LOCA Cold Leg
Random Forest	Steady State	0	0	0	0	0
	Rod Withdrawal	100	0	0	0	0
	Steam Leak from Pressurizer	0	100	0	0	0
	Loss of Flow	0	0	100	0	0
	LOCA Hot Leg	0	0	0	98.56	1.44
	LOCA Cold Leg	0	0	0	0	100
Gradient Boosting	Steady State	0	0	0	0	0
	Rod Withdrawal	100	0	0	0	0
	Steam Leak from Pressurizer	0	100	0	0	0
	Loss of Flow	0	0	100	0	0
	LOCA Hot Leg	0	0	0	98.28	1.72
	LOCA Cold Leg	0	0	0	0	100
Logistic Regression	Steady State	0	0	0	0	0
	Rod Withdrawal	100	0	0	0	0
	Steam Leak from Pressurizer	0	100	0	0	0
	Loss of Flow	0	0	100	0	0
	LOCA Hot Leg	0	0	0	96.56	3.44
	LOCA Cold Leg	0	0	0	1	99

The significance of various features has been evaluated for both the Random Forest, Gradient Boosting, and Logistic Regression methods. The Gini impurity was

considered and utilized for determining importance and significance of the features through Python, a comparison for feature types with respect to importance levels relying on Gini impurity, has been assessed in Table 6.3. While Random Forest and Logistic Regression methods rely on mass flow rate, temperature, and pressure features, Gradient Boosting relies heavily on mass flow rate feature. However, almost 85% of explanation power comes from mass flow rate, pressure, and temperature features for all methods, while water levels also displays some role.

Table 6.3: Feature type and importance for methods used for main transient identification

Feature Type	Random Forest	Logistic Regression	Gradient Boosting	Feature Count
Mass Flow	49.72	49.26	75.49	34
Pressure	22.89	23.67	6.52	25
Temperature	14.07	14.89	5.58	23
Water Level	5.62	4.76	2.77	5
Volume Saturation Temperature	3.89	2.59	1.86	1
Saturation Temperature	2.33	2.32	0.58	1
Void Fraction	0.95	1.95	6.23	1
Power	0.53	0.56	0.96	1

Top 10 features for Random Forest and corresponding importance of those features for other methods is displayed at Figure 6.1. Mass flow rate and temperature values for reactor core features play a significant role as expected especially for Gradient Boosting. However, Logistic Regression reveals a close importance distribution for features listed in Figure 6.1.

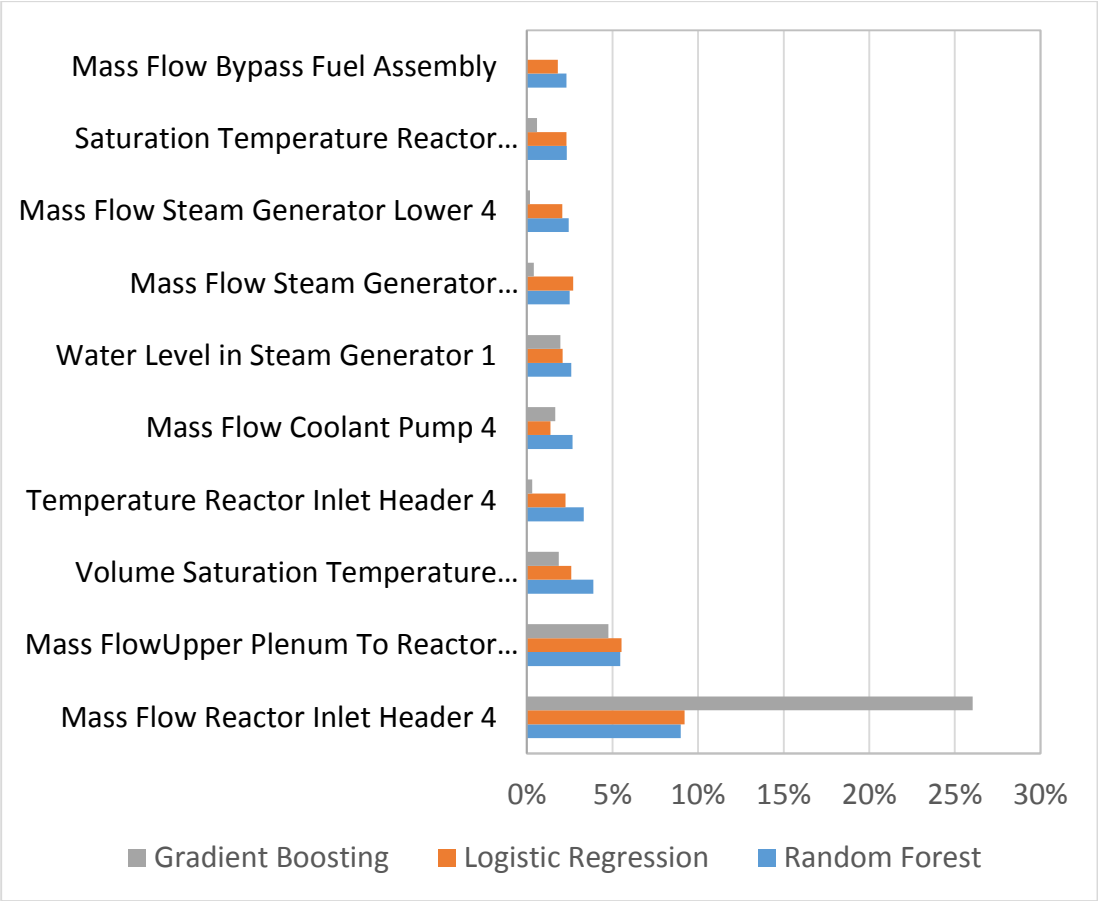

Figure 6.1: Importance of features for Gradient Boosting, Logistic Regression and Random Forest methods for main transient identification

The results of the learning curve are presented in Table 6.4 and Figure 6.2. Table 6.4 illustrates that, with respect to training accuracy, the models exhibit a strong fit, with nearly all models achieving 100%. In terms of validation accuracy, although the learning curve gradients are similar, the Random Forest method achieves 94 % accuracy with 2941 samples, surpassing the performance of the other methods.

Table 6.4: Learning curve data for Random Forest, Logistic Regression, and Gradient Boosting methods

Method	Sample Size	Training Accuracy (%)	Validation Accuracy (%)
Random Forest	980	99.00	33.00
	1960	99.00	66.00
	2941	100.00	94.00
Logistic Regression	980	93.00	31.00
	1960	96.00	63.00
	2941	97.00	88.00
Gradient Boosting	980	99.00	33.00
	1960	99.00	64.00
	2941	100.00	91.00

Figure 6.2 demonstrates that validation accuracy tends to increase almost linearly with an increase in sample size, whereas training accuracy remains constant at 99 % across all sample sizes. The variability of the method is depicted through shaded regions, representing the standard deviation above and below the mean for all cross-validation instances. An increase in variability around the training score curve suggests that the model may be experiencing bias-related errors. Conversely, greater variability around the validated score implies that the model may be affected by variance-related errors. As such while adding more training data increases accuracy, the model tends to risk of over fitting by becoming too sensitive to minor fluctuations. The graphs in Figure 6.2 shows that there is not such risk in validation.

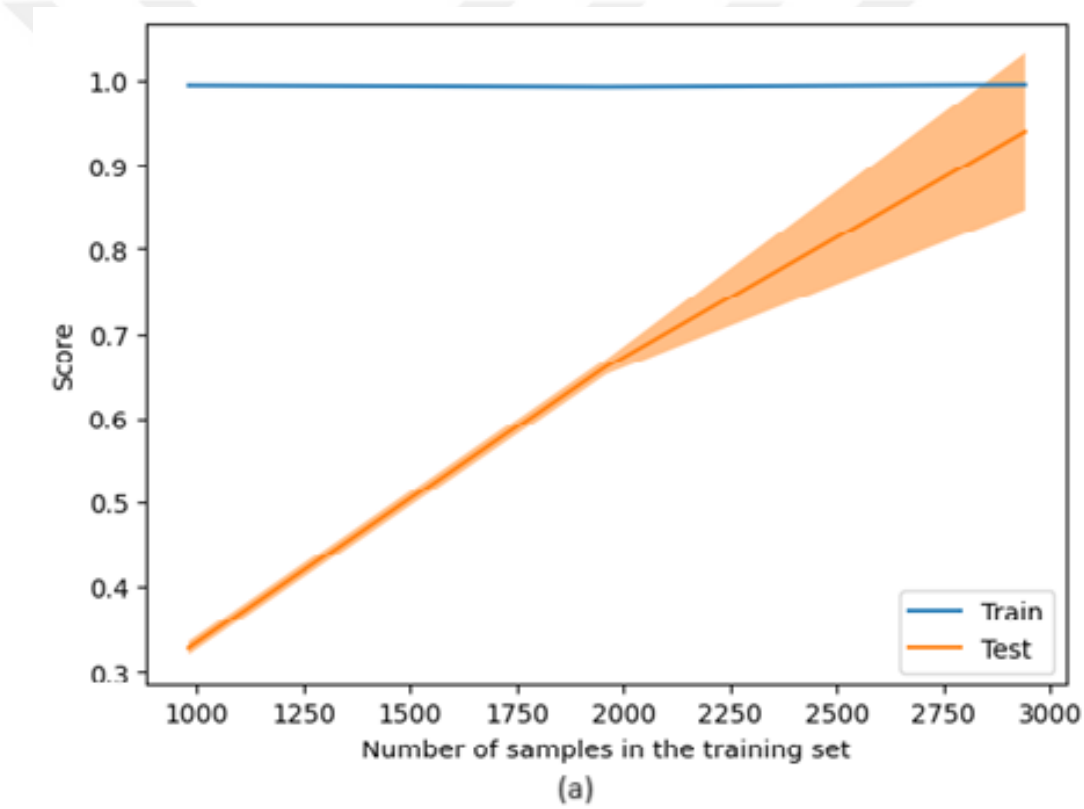

Figure 6.2: Learning curve visualization for ML methods Random Forest (a), Logistic Regression (b), Gradient Boosting (c)

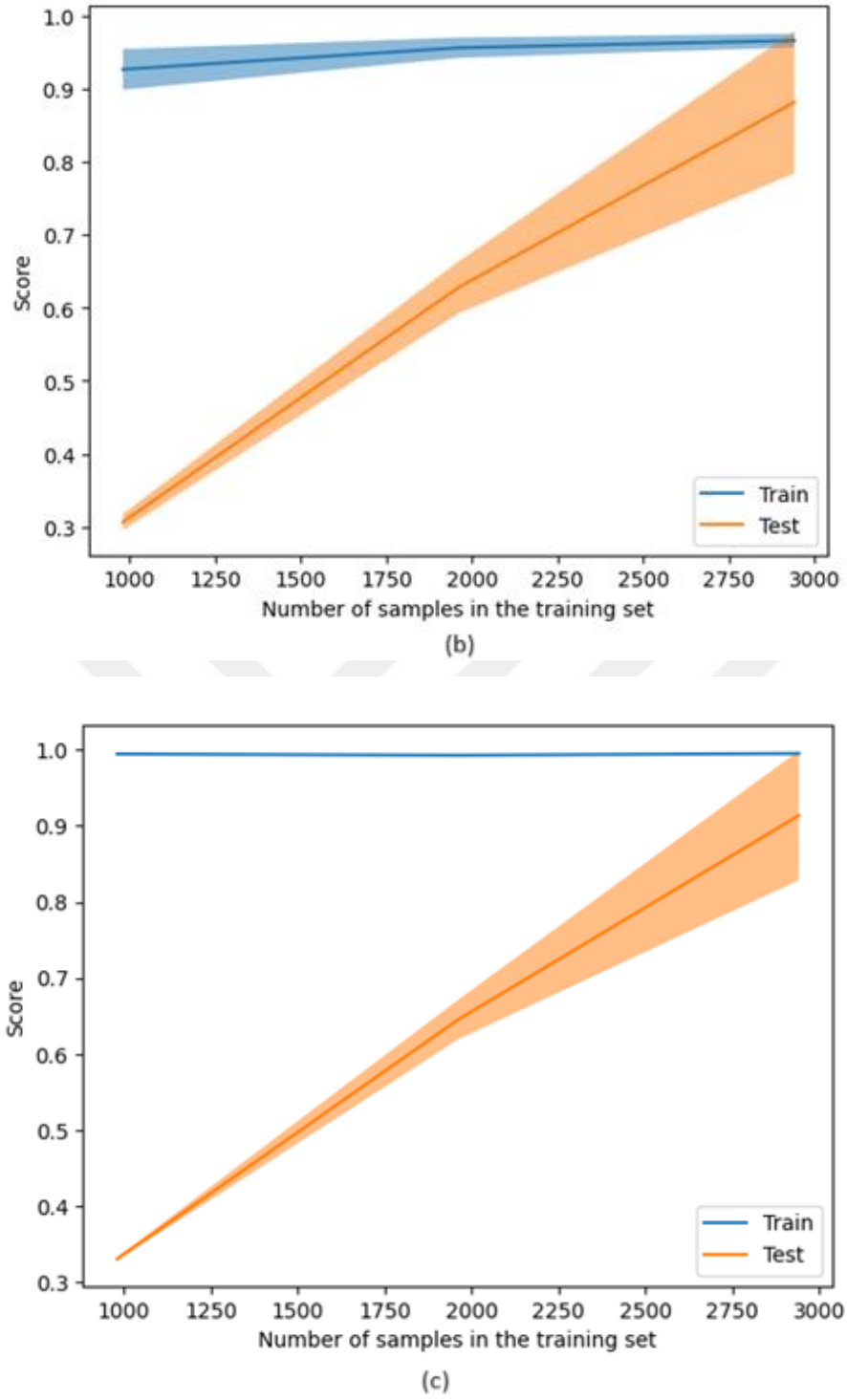

Figure 6.2: Learning curve visualization for ML methods Random Forest (a), Logistic Regression (b), Gradient Boosting (c) (Continue)

6.2 Results for Sub-Scenario Identificaiton

While the performance for main transient identificaiton is satisfactory, the real scope of this work is about distinguishing between levels of main transients. Three approaches were proposed in this concept, one-step approach that utilizes models

directly for all sub-scenarios , two-steps approach, which after utilizing the main transient model, which has a prediction rate of 99.51% with Random Forest method, training and executing a model for every main transient to distinguish between sub-scenarios within their respective main transients and lastly Grouped Sub-Scenarios Approach which defines new scenarios by grouping sub-scenarios with respect to their confusion levels and predicts newly formed scenarios.

6.2.1 Results for one-step approach

In one-step approach, KNN, Decision Tree Classifier, Random Forest Classifier, Gradient Boosting, Logistic Regression, SVM and MLP ML methods directly applied to all data for 53 sub-scenarios. The resulting accuracy values are provided in Table 6.5. Although KNN method provided the highest accuracy, 74.66 % is not acceptable for nuclear applications.

Table 6.5: Accuracy of ML methods used in one-step approach

Method	Accuracy (%)
K-Nearest Neighbor	74.66
Random Forest	55.63
Gradient Boosting	52.33
Decision Trees	49.32
Logistic Regression	44.47
Support Vector	36.02
Naïve Bayes	22.04

Table 6.6 indicates that for steady-state and loss of flow scenarios, KNN method does not have any confusion however for all other scenarios there is confusion.

Table 6.6: Accuracy of K-Nearest Neighbor in one-step approach

Transient	Accuracy (%)
Steady State	100
Rod Withdrawal	75
Steam Leak from Pressurizer	83.33
Loss of Flow	100
LOCA Hot Leg	63.79
LOCA Cold Leg	76.61

Considering the confusion matrix for LOCA cold leg provided in Figure 6.3, KNN method not only totally misclassifies for 85% level of the transient, but also misclassifies the main transient for LOCA hot leg denoted as H15 and H90 in Figure 6.3.

While confusion matrix for LOCA hot leg, which is provided in Figure 6.4, does not misclassify in terms of main transient, totally fails at classifying for 85% sub-scenario and has a worse classification performance than LOCA cold leg in terms of accuracy.

For rod withdrawal and steam leak from pressurizer confusion matrixes that are represented in Figure 6.5, while accuracy performance is 75 % and 83.33 % respectively, confusion is concentrated in the order of one level below.

It is clear that one-step model has difficulty in distinguishing between levels of transient sub-scenarios. This is because of the fact that nuclear system response is complicated and some transients affect the same features that are used for AI training.

Cold Leg Loca																			
Sub Scenario	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**	**65**	**70**	**75**	**80**	**85**	**90**	**H15**	**H90**
LocaColdLeg05	100%																			
LocaColdLeg10		75%	15%																10%	
LocaColdLeg15			60%	40%																
LocaColdLeg20				50%	50%															
LocaColdLeg25					55%	25%	5%					5%		10%						
LocaColdLeg30					5%	60%	35%													
LocaColdLeg35						5%	95%													
LocaColdLeg40							5%	95%												
LocaColdLeg45							5%		95%											
LocaColdLeg50										95%					5%					
LocaColdLeg55										5%	95%									
LocaColdLeg60											5%	90%		5%						
LocaColdLeg65													95%	5%						
LocaColdLeg70												40%		55%	5%					
LocaColdLeg75															100%					
LocaColdLeg80															9%			91%		
LocaColdLeg85														9%	9%			18%		64%
LocaColdLeg90																5%		95%		

Figure 6.3: Confusion matrix for one-step LOCA cold leg sub-scenarios in one-step approach

Hot Leg Loca																	
Sub Scenario	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**	**50**	**55**	**60**	**65**	**70**	**75**	**80**	**85**	**90**
LocaHotLeg05	100%																	
LocaHotLeg10		90%	10%															
LocaHotLeg15			25%	75%														
LocaHotLeg20				40%	60%													
LocaHotLeg25					80%													20%
LocaHotLeg30						60%	35%	5%										
LocaHotLeg35						5%	15%	35%			45%							
LocaHotLeg40							5%	15%			10%	55%	5%	10%				
LocaHotLeg45								5%	20%	75%								
LocaHotLeg50									5%	60%			15%	20%				
LocaHotLeg55											35%	60%		5%				
LocaHotLeg60												95%						5%
LocaHotLeg65													90%	10%				
LocaHotLeg70													5%	95%				
LocaHotLeg75															100%			
LocaHotLeg80													5%			95%		
LocaHotLeg85													13%			50%		38%
LocaHotLeg90																5%		95%

Figure 6.4: Confusion matrix for one-step LOCA hot leg sub-scenarios in one-step approach

Rod Withdrawal									
Sub Scenario	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**
RodWithdrawal05	100%								
RodWithdrawal10	80%	20%							
RodWithdrawal15		70%	30%						
RodWithdrawal20			30%	70%					
RodWithdrawal25				25%	75%				
RodWithdrawal30					5%	95%			
RodWithdrawal35						5%	95%		
RodWithdrawal40							5%	95%	
RodWithdrawal45							5%		95%

Steam Leak			
Sub Scenario	**1**	**12**	**123**
PrezLoss1	100%		
PrezLoss12		55%	45%
PrezLoss123		5%	95%

Figure 6.5: Confusion matrix for one-step rod withdrawal and steam leak from pressurizer sub-scenarios in one-step approach

6.2.2 Results for two-steps approach

After determining the main transients, next step in two-steps approach is to establish methods for detecting sub-scenarios within the main transients. KNN, Decision Tree Classifier, Random Forest Classifier, Gradient Boosting, Logistic Regression, SVM, and MLP methods were applied, executing hyper parameter tuning runs. The same principle of 80 %-20 % data training-validation set was applied. In addition, for method optimization, feature reduction was applied with respect to feature importance values in Figure 6.1. Overall accuracy metric for sub-scenario identification, which is the combined accuracy of rod withdrawal, steam leak from pressurizer , loss of flow, LOCA

hot leg, and LOCA cold leg scenarios in Table 6.7, calculated by dividing total true positive classifications divided by entire validation set number.

Table 6.7: Accuracy of ML methods used for sub-scenario identification

Method	Rod Withdrawal	Steam Leak from Pressurizer	Loss Of Flow	Loca Hot Leg	Loca Cold Leg	Overall Accuracy
K-Nearest Neighbor	100	100	100	78.74	82.75	86.83
Random Forest	22	48	100	56.32	62.87	55.39
Gradient Boosting	16.11	66.67	100	51.72	59.36	52.67
Decision Trees	22	40	77.50	53.74	45.61	46.40
Logistic Regression	18	66.67	100	30.75	60.53	46.18
Support Vector	11	48	97.50	26.72	61.70	42.63
Multi-Layer Perceptron	32	33.33	100	37.36	16.67	34.12
Naïve Bayes	11	66.67	97.50	11.49	37.72	30.38

K-Nearest Neighbor scored stellar for steam leak from pressurizer and rod withdrawal scenarios while performing close to mean of 80 % for LOCA cold leg and hot leg scenarios. The confusion matrix for LOCA hot leg and cold leg is given in Figure 6.6 and Figure 6.7 respectively since there is no confusion in rod withdrawal, steam leak from pressurizer , and loss of flow sub-scenarios.

For LOCA cold leg sub-scenarios, Figure 6.6 indicates that even though general accuracy performance is close to 83 %, cold leg 20, 30 and 85 sub-scenarios are confused with sub-scenarios, especially one level below and above in intensity.

Cold Leg Loca Sub Scenario	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
LocaColdLeg05	80%	20%																
LocaColdLeg10		85%	15%															
LocaColdLeg15			85%	15%														
LocaColdLeg20				35%	65%													
LocaColdLeg25					95%													5%
LocaColdLeg30						30%	65%	5%										
LocaColdLeg35							95%			5%								
LocaColdLeg40								95%			5%							
LocaColdLeg45									95%				5%					
LocaColdLeg50										95%					5%			
LocaColdLeg55											95%					5%		
LocaColdLeg60											5%	95%						
LocaColdLeg65												5%	95%					
LocaColdLeg70														95%				5%
LocaColdLeg75															100%			
LocaColdLeg80						18%									9%	73%		
LocaColdLeg85						18%									9%	55%	18%	
LocaColdLeg90															5%			95%

Figure 6.6: Confusion matrix for LOCA cold leg sub-scenarios in two-steps approach

For LOCA hot leg sub-scenarios, even though general accuracy performance is close to 80 %, differing from LOCA cold leg sub-scenario confusion, hot leg 30, 35 and 45 sun-scenarios are confused with scenarios of varying intensity as seen in Figure 6.7. For LOCA hot leg 85 sub-scenario, the model misses out completely confusing it with LOCA 80 hot leg sub-scenario.

Hot Leg Loca Sub Scenario	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
LocaHotLeg05	100%																	
LocaHotLeg10		95%	5%															
LocaHotLeg15			95%					5%										
LocaHotLeg20				85%	15%													
LocaHotLeg25					100%													
LocaHotLeg30						40%	15%		25%	15%		5%						
LocaHotLeg35							10%	15%			75%							
LocaHotLeg40								50%		20%	5%	25%						
LocaHotLeg45									40%	55%			5%					
LocaHotLeg50										95%					5%			
LocaHotLeg55											95%					5%		
LocaHotLeg60												95%						5%
LocaHotLeg65												5%	95%					
LocaHotLeg70													5%	95%				
LocaHotLeg75													5%		95%			
LocaHotLeg80													5%			95%		
LocaHotLeg85															13%	88%		
LocaHotLeg90															5%	5%		90%

Figure 6.7: Confusion matrix for LOCA hot leg sub-scenarios in two-steps approach

Hyper parameter optimization and feature reduction on training data for K-Nearest Neighbor method is given in Table 6.8 per transient. K stands for number of neighbors

to be trained for K-Nearest Neighbors method. Table 6.8shows that the usage of all features is not necessary to get the best results.

Table 6.8: Parameters for K-Nearest Neighbor method in two-steps approach

Transient	K	Feature Selection
Rod Withdrawal	3	Top 20
Steam Leak from Pressurizer	2	Top 15
Loss of Flow	2	All
LOCA Hot Leg	2	Top 14
LOCA Cold Leg	2	Top 10

The results of the two-steps approach for transient and sub-scenario detection showed that the most successful models are Random Forest for the first step and K-Nearest Neighbors for the second step. The combined accuracy for Random Forest/K-Nearest Neightbors two-steps approach for all transients together with total accuracy is given in Table 6.9 as 86.44 %. The combined accuracy is calculated by using the accuracy of Random Forest method given in Table 6.1 for main transient identification and accuracy of KNN given in Table 6.7 for sub-scenario identification.

Table 6.9: Combined accuracy of Random Forest/K-Nearest Neighbor two-steps approach

Transient	Final Accuracy (%)
Steady State	100
Rod Withdrawal	100
Steam Leak from Pressurizer	100
Loss of Flow	100
LOCA Hot Leg	78
LOCA Cold Leg	82.75
Total Accuracy	86.44

6.2.3 Results for grouped sub-scenarios approach

With respect to Figure 6.10 and Figure 6.11, most of the confusion levels of the main transients is mostly in the order of one level, namely 5% above or below, which suggests with grouping of sub-scenarios, a higher level of accuracy is possible. As a result of this idea, a new grouping of the sub-scenarios with 19 groups was performed and presented in Table 6.10.

Table 6.10: Grouped sub-scenarios for the transients

Main Transient	Sub-Scenario
Steady-State	Steady-State
Rod Withdrawal	RodWithdrawal_05-15
Rod Withdrawal	RodWithdrawal_20-30
Rod Withdrawal	RodWithdrawal_35-45
Steam Leak from Pressurizer	PrezLoss_1
Steam Leak from Pressurizer	PrezLoss_12
Steam Leak from Pressurizer	PrezLoss_123
Loss of Flow	LossofFlow_4
Loss of Flow	LossofFlow_34
Loss of Flow	LossofFlow_234
Loss of Flow	LossofFlow_1234
LOCA Hot Leg	LocaHotLeg_05-25
LOCA Hot Leg	LocaHotLeg_30-50
LOCA Hot Leg	LocaHotLeg_55-75
LOCA Hot Leg	LocaHotLeg>=80
LOCA Cold Leg	LocaColdLeg_05-25
LOCA Cold Leg	LocaColdLeg_30-50
LOCA Cold Leg	LocaColdLeg_55-75
LOCA Cold Leg	LocaColdLeg>=80

The accuracy performance of grouped one-step approach is shown in Table 6.11. It is seen that there is an increase in accuracy for K-Nearest Neighbor method to 92.33 % from 86.83 % and 74.66 % for the same method in two-steps and one-step approaches, respectively. On the other hand, it should be noted that while this is an accuracy gain with respect to loss of sensitivity in distinguishing level of transients.

Table 6.11: Accuracy of ML methods used for grouped one-step approach

Method	**Accuracy**
K-Neighbor	92.33
Gradient Boosting	88.06
Random Forest	87.57
Logistic Regression	79.22
Decision Trees	76.50
Support Vector	66.02
Naïve Bayes	43.30

7. CONCLUSUIONS AND RECOMMENDATIONS

The transient identification of VVER-1000 NPP was performed by considering three approaches each of which uses AI. In one-step approach, data for all the sub-scenarios were used directly to identify transient sub-scenarios. In the two-steps approach, the main transients were identified first and then the sub-scenario identification was performed. In grouped one-step approach, with the guidance of confusion matrix of one-step approach, the sub-scenarios were grouped to increase the accuracy of the predictions.

The best performance of identification in one-step approach was achieved with KNN method which resulted in 74.66 % accuracy. This result shows that if all transient sub-scenarios were considered in one training method, there is a lot of confusion due to complex nature of nuclear system. KNN, being a simpler ML method than others, outperformed all models, showcasing separation of high scenario numbers is challenging for ML models.

In two-steps approach, main transient detection with Random Forest method yielded the best result in terms of accuracy, precision, recall, and F1-score matrices, only slightly confusing LOCA hot leg transient with 1.44 %. This provided the sub-scenario identification almost 100% accurate main transient base to optimize on. Sub-scenario identification with KNN method delivered 100% accuracy for rod withdrawal, steam leak from pressurizer, and loss of flow transients, providing the reactor operator an accurate understanding on the level of the transient. However, for LOCA transients, the method had a combined accuracy of approximately 80%, which made the sub-scenario model usable for these transients, yielding a total of 86.44 % accuracy for two-steps approach. Since LOCA scenarios affect the same parameters regardless of their intensity, while Random Forest method, a highly stochastic model, performs the best in main transient detection, KNN the fourth successful method in main transient detection, performed strongest in second fold detection, per say transient intensity detection, proof casing the better performance of simple method in one-step model.

Analyzing of confusion matrices of sub-scenarios of two-steps approach, it was observed that most of the confused scenarios are within the range of 1 or 2 closest intensity level, hence with new grouping with respect to that range resulted in 19 grouped sub-scenarios. This yielded a far better accuracy among all approaches with 92.04 % for KNN method. On the other hand, due to grouping, there is a loss of sensitivity in this approach. However this time, due to decreasing sensitivity check for scenarios, Random Forest and Gradient Boosting methods performed 87.57% and 88.06% getting closer to KNN, trend being as number of scenarios and sensitivity decrease, other ML methods do significantly better.

For all approaches, the validation process was completed in 2-3 seconds, indicating that the approaches is sufficiently efficient and responsive for real-time monitoring applications.

While two-steps and Grouped Sub-Scenarios approaches' performance are acceptable with respect to high performance demand of nuclear field, number of features to attain the accuracy of models is 91 which is a number prone to failure in case of loss of signal of any nuclear reactor instrument tally.

A real application of the developed AI model can be demonstrated with feeding actual data output of a VVER-1000 reactor to the extent of the features and transient experiment data that can be provided and with a condensed trained instance of the model. Since the data will be real time, outlier removal and normalization must take place again.

For future work, to mitigate problems in data stream necessity from such a wide array of features and optimize for further fewer features and higher accuracy, deep learning methods can be implemented while incorporating determination of not only transient but the exact locaiton in reactor nodalization.

REFERENCE

[1]. **Yin, W., Xia, H., Huang, X., Zhang, J., & Miyombo, M. E.** (2023). A fault diagnosis method for nuclear power plant rotating machinery based on adaptive deep feature extraction and multiple support vector machines. *Progress in Nuclear Energy, 164*, 104862. https://doi.org/10.1016/j.pnucene.2023.104862

[2]. **Young, A. T., Aylward, W., Murray, P., West, G. M., & McArthur, S. D. J.** (2019). Automatic anomaly detection in fuel grab load trace data using a knowledge-based system vs. multiple deep autoencoders. https://pureportal.strath.ac.uk/en/publications/automatic-anomaly-detection-in-fuel-grab-load-trace-data-using-a-

[3]. **Hammad, I., Simpson, R., Tsague, H. D., & Hall, S.** (2021). "Using Deep Learning to Automate the Detection of Flaws in Nuclear Fuel Channel UT Scans," in *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 1, pp. 323-329, Jan. 2022, doi: 10.1109/TUFFC.2021.3112078

[4]. **Guo, Z., Wu, Z., Liu, S., Ma, X., Wang, C., Yan, D., & Niu, F**. (2019). Defect detection of nuclear fuel assembly based on deep neural network. *Annals of Nuclear Energy, 137*, 107078. https://doi.org/10.1016/j.anucene.2019.107078

[5]. **B. Zhuang, A. Arcaro, B. Gencturk, R. Meyer, A. Oberai, A. Sinkov, M.Good. (2025)**. Non-destructive evaluation and machine learning methods for inspection of spent nuclear fuel canisters: A state-of-the-art review, *Progress in Nuclear Energy*, Volume185, 105697, ISSN0149-1970, https://doi.org/10.1016/j.pnucene.2025.105697.

[6]. **B. Zhuang, A. Arcaro, B. Gencturk, R. Ghanem. (2024).** Machine learning-aided damage identification of mock-up spent nuclear fuel assemblies in a sealed dry storage canister, *Engineering Applications of Artificial Intelligence*,

Volume 128, 107484, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2023.107484.

[7]. **Pereira, C. M., Schirru, R., Gomes, K. J., & Cunha, J. L.** (2017). Development of a mobile dose prediction system based on artificial neural networks for NPP emergencies with radioactive material releases. *Annals of Nuclear Energy, 105,* 219–225. https://doi.org/10.1016/j.anucene.2017.03.017

[8]. **Manimegalai, P., Kumar, R. S., Valsalan, P., Dhanagopal, R., Raj, P. T. V., & Christhudass, J.** (2022). 3D Convolutional Neural Network Framework with Deep Learning for Nuclear Medicine. Scanning. ,9640177. doi: 10.1155/2022/9640177.

[9]. **Desterro, F. S., Santos, M. C., Gomes, K. J., Heimlich, A., Schirru, R., & Pereira, C. M.** (2019). Development of a Deep Rectifier Neural Network for dose prediction in nuclear emergencies with radioactive material releases. *Progress in NuclearEnergy*, *118*,103110 https://doi.org/10.1016/j.pnucene.2019.103110

[10]. **L.M.N. Barcellos, P.C.R. Silveira, A.S. Nicolau, R. Schirru, C.M.N.A. Pereira, (2023).** Dynamic prediction of spatial dose rate distribution during nuclear severe accidents by means of active machine learning and mobile sensors,*Nuclear Engineering and Design*,Volume 414,2023,12609,ISSN 0029-5493,https://doi.org/10.1016/j.nucengdes,112609.

[11]. **C. Guo, X. Li, Z. Yan, L. Chen, B. Tang, W. Zeng, (2025)**. Prediction methodology of air absorbed dose rates for Chinese cities with deep learning models, *Journal of Environmental Radioactivity*, Volume 286, ISSN 0265-931X, https://doi.org/10.1016/j.jenvrad.2025.107685.

[12]. **Ding, M., Zhou, X., Zhang, H., Bian, H., & Yan, Q. (2021).** *A review of the development of nuclear fuel performance analysis and codes for PWRs.Annals of Nuclear Energy, 163, 108542 https://doi.org/10.1016/j.anucene.2021.108542*

[13]. **Dong, B., Xiao, W., Yin, J., & Wang, D**. (2019). Detection of fuel failure in pressurized water reactor with artificial neural network. *Annals of Nuclear Energy, 140,* 107104. https://doi.org/10.1016/j.anucene.2019.107104

[14]. **Saeed, A., & Rashid, A.** (2020). Development of Core Monitoring System for a Nuclear Power Plant using Artificial Neural Network Technique. *Annals of Nuclear Energy*, *144*, 107513. https://doi.org/10.1016/j.anucene.2020.107513

[15]. **Xia, H., Li, B., & Liu, J.** (2014). Research on intelligent monitor for 3D power distribution of reactor core. *Annals of Nuclear Energy*, *73*, 446–454. https://doi.org/10.1016/j.anucene.2014.07.033

[16]. **Filipe S.M. Desterro, Victor H.C. Pinheiro, Cláudio M.N.A. Pereira, Roberto Schirru** (2023). Prediction of LOCA's break size and location based on random forest and Multi Tasking Deep Neural Network,Nuclear Engineering and Design, Volume415, ISSN 0029-5493, https://doi.org/10.1016/j.nucengdes

[17]. **Sadeghi, K., Ghazaie, S. H., Sokolova, E., Cammi, A., Arab, H. R., & Usta, S.** (2023). A set of transient correlations for fast and unprotected loss of flow accident in VVER-1000 reactor using single-heated channel approach and gene expression programming. *Annals of Nuclear Energy, 183*, 109650. https://doi.org/10.1016/j.anucene.2022.109650

[18]. **Hadad, K., Pourahmadi, M., & Majidi-Maraghi, H.** (2011). Fault diagnosis and classification based on wavelet transform and neural network. *Progress in Nuclear Energy*, *53*(1), 41–47. https://doi.org/10.1016/j.pnucene.2010.09.006

[19]. **Salmassian, B., Rabiee, A., Nematollahi, M. R., & Pirouzmand, A.** (2023). Diagnosing core local flow blockages in a VVER-1000/446 reactor using ex-core detectors and neural networks. *Progress in Nuclear Energy, 161*, 104736. https://doi.org/10.1016/j.pnucene.2023.104736

[20]. **Zehtabvar, Mehrnaz, Kazem Taghandiki, Nahid Madani, Dariush Sardari, and Bashir Bashiri. (**2024) "A Review on the Application of Machine Learning in Gamma Spectroscopy: Challenges and Opportunities" Spectroscopy Journal 2, no. 3: 123-144. https://doi.org/10.3390/spectroscj2030008

[21]. **Sahiner, H.** (2017). *Gamma spectroscopy by artificial neural network coupled with MCNP* [Doctoral dissertation]. University of Missouri.

[22]. **Souza, R. M. G., & Moreira, J. M**. (2006). Neural network correlation for power peak factor estimation. *Annals of Nuclear Energy*, *33*(7), 594–608. https://doi.org/10.1016/j.anucene.2006.02.007

[23]. **Babazadeh, D., Boroushaki, M., & Lucas, C.** (2009). Optimization of fuel core loading pattern design in a VVER nuclear power reactors using Particle Swarm Optimization (PSO). *Annals of Nuclear Energy*, *36*(7), 923–930. https://doi.org/10.1016/j.anucene.2009.03.007

[24]. **Misak, J.** (2024). History, specific design features, and evolution of VVER reactors. In *Elsevier eBooks* (pp. 57–91). https://doi.org/10.1016/b978-0-323-99880-2.00004-7

[25]. **Fekete, T.** *(2016). Review of pressurized thermal shock studies of large scale reactor pressure vessels in Hungary. Frattura Ed Integrità Strutturale, 10(36), 99–111. https://doi.org/10.3221/igf-esis.36.10*

[26]. **INTERNATIONAL ATOMIC ENERGY AGENCY** (2024). Nuclear Power Reactors in the World, Reference Data Series No. 2, IAEA, Vienna

[27]. **VVER Working Group.** (2019). Preliminary analyses & comparison of design differences between VVER-1000 reactors.

[28]. **Tabadar, Z., Aghajanpour, S., Jabbari, M., Khaleghi, M., & Hashemi-Tilehnoee, M.** (2018). Thermal-hydraulic analysis of VVER-1000 residual heat removal system using RELAP5 code, an evaluation at the boundary of reactor repair mode. *Alexandria Engineering Journal*, *57*(3), 1249–1259. https://doi.org/10.1016/j.aej.2017.03.044

[29]. **S. A. Aljasar, Y. V. Stogov, M. Aish**. 2021 "Comparison of the neutronic calculations of the cells of VVER-1000 and PWR reactors using the GETERA code", *AM*, vol. 9, no. 3, pp. 1079–1097

[30]. **Khan, A. H., Hossain, S., Hasan, M., Islam, M. S., Rahman, M. M., & Kim, J.** (2022). Development of an optimized thermodynamic model for VVER-1200 reactor-based nuclear power plants using genetic algorithm. *Alexandria Engineering Journal*, *61*(11), 9129–9148. https://doi.org/10.1016/j.aej.2022.02.052

[31]. **ROSATOM.** (n.d.). *The VVER today: Evolution, design, safety*.

[32]. **Alpaydin, E.** (2016). *Machine Learning: The New AI*. MIT Press.

[33]. **Taye, M. M. (2023)**. Understanding of Machine Learning with Deep Learning: Architectures, Workflow, *Applications and Future Directions*. Computers, 12(5), 91. https://doi.org/10.3390/computers12050091

[34]. **Cunningham, P. & Delany, S. J.** (2021). K-Nearest Neighbour Classifiers – a tutorial. *ACM Computing Surveys*, *54*(6),1–25. https://doi.org/10.1145/3459665

[35]. **Ortega-Arranz, H., Llanos, D. R., & Gonzalez-Escribano, A**. (2015). The Shortest-Path problem. In *Synthesis lectures on theoretical computer science*. https://doi.org/10.1007/978-3-031-02574-7

[36]. **Koufos, K., Dhillon, H. S., Dianati, M., & Dettmann, C. P.** (2021)**.** On the k Nearest-Neighbor Path Distance from the Typical Intersection in the Manhattan Poisson Line Cox Process. *IEEE Transactions on Mobile Computing*, 1. https://doi.org/10.1109/tmc.2021.3108067

[37]. **Shapcott, Z.** (2024). An Investigation into Distance Measures in Cluster Analysis. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2404.13664

[38]. **Halder, R. K., Uddin, M. N., Uddin, M. A., Aryal, S., & Khraisat, A**. (2024). Enhancing K-nearest neighbor algorithm: a comprehensive review and performance analysis of modifications. *Journal of Big Data*, *11*(1). https://doi.org/10.1186/s40537-024-00973-y

[39]. **Blockeel, H., Devos, L., Frénay, B., Nanfack, G., & Nijssen, S**. (2023). Decision trees: from efficient prediction to responsible AI. *Frontiers in Artificial Intelligence*, *6*. https://doi.org/10.3389/frai.2023.1124553

[40]. **Biau, G., Scornet, E.** (2016) A random forest guided tour. *TEST* **25**, 197–227 https://doi.org/10.1007/s11749-016-0481-7

[41]. **Racheal, S., Liu, Y., & Ayodeji, A.** (2022). Evaluation of optimized machine learning models for nuclear reactor accident prediction. *Progress in Nuclear Energy*, *149*, 104263. https://doi.org/10.1016/j.pnucene.2022.104263

[42]. **Palimkar, P., Shaw, R.N., Ghosh, A.** (2022). *Machine Learning Technique to Prognosis Diabetes Disease: Random Forest Classifier Approach*. In:

Bianchini, M., Piuri, V., Das, S., Shaw, R.N. (eds) Advanced Computing and Intelligent Technologies. Lecture Notes in Networks and Systems, vol 218. Springer, Singapore. https://doi.org/10.1007/978-981-16-2164-2_19

[43]. **Breiman,L.** (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/a:1010933404324

[44]. **Biau, G., & Cadre, B.** *(2017). Optimization by gradient boosting. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1707.05023*

[45]. **Al-Haddad, L. A., Jaber, A. A., Hamzah, M. N., & Fayad, M. A.** (2023). Vibration-current data fusion and gradient boosting classifier for enhanced stator fault diagnosis in three-phase permanent magnet synchronous motors. *Electrical Engineering*, *106*(3), 3253–3268. https://doi.org/10.1007/s00202-023-02148-z

[46]. **Joshi, R. D., & Dhakal, C. K.** (2021). Predicting Type 2 Diabetes Using Logistic Regression and Machine Learning Approaches. *International Journal of Environmental Research and Public Health*, *18*(14), 7346. https://doi.org/10.3390/ijerph18147346

[47]. **Guerrero, M. C., Parada, J. S., & Espitia, H. E.** (2021). EEG signal analysis using classification techniques: Logistic regression, artificial neural networks, support vector machines, and convolutional neural networks. *Heliyon*, *7*(6), e07258. https://doi.org/10.1016/j.heliyon.2021.e07258

[48]. **Ziyad, S. R., Liyakathunisa, Aljohani, E., & Saeed, I. A.** (2023). Diagnosis of autism spectrum disorder by imperialistic competitive algorithm and logistic regression classifier. *Computers, Materials & Continua*, *77*(2), 1515–1534. https://doi.org/10.32604/cmc.2023.040874

[49]. **Abdullah, D. M., & Abdulazeez, A. M.** (2021). Machine Learning Applications based on SVM Classification A Review. *Qubahan Academic Journal*, *1*(2), 81–90. https://doi.org/10.48161/qaj.v1n2a50

[50]. **Roman, I., Santana, R., Mendiburu, A., & Lozano, J. A.** (2020). In-depth analysis of SVM kernel learning and its components. *Neural Computing and Applications*, *33*(12), 6575–6594. https://doi.org/10.1007/s00521-020-05419-z

[51]. **Bahtiyar, H., Soydaner, D., & Yüksel, E.** (2022). Application of multilayer perceptron with data augmentation in nuclear physics. *Applied Soft Computing*, *128*, 109470. https://doi.org/10.1016/j.asoc.2022.109470

[52]. **El-Sefy, M., Yosri, A., El-Dakhakhni, W., Nagasaki, S., & Wiebe, L.** (2021). Artificial neural network for predicting nuclear power plant dynamic behaviors. *Nuclear Engineering and Technology*, *53*(10), 3275–3285. https://doi.org/10.1016/j.net.2021.05.003

[53]. **Y., Lin, T., & Fan, Z.** (2021). Development and outlook of advanced nuclear energy technology. *Energy Strategy Reviews*, *34*, 100630. https://doi.org/10.1016/j.esr.2021.100630

[54]. **Guliyev, H., & Tatoğlu, F. Y.** (2025). When Did the Labor Force Replace Energy?—Evidence from the Panel Data Model with Structural Breaks. *Journal of the Knowledge Economy*. https://doi.org/10.1007/s13132-025-02650-8

[55]. **Bataineh, A. A., Kaur, D., & Jalali, S. M. J.** (2022). Multi-Layer Perceptron Training optimization using nature inspired computing. *IEEE Access*, *10*, 36963–36977. https://doi.org/10.1109/access.2022.3164669

[56]. **Sharma, R., Kim, M., & Gupta, A.** (2021). Motor imagery classification in brain-machine interface with machine learning algorithms: Classical approach to multi-layer perceptron model. *Biomedical Signal Processing and Control*, *71*, 103101. https://doi.org/10.1016/j.bspc.2021.103101

[57]. **Kiliçarslan, S., Adem, K., & ÇeliK, M.** (2021). An overview of the activation functions used in deep learning algorithms. *Journal of New Results in Science*, *10*(3), 75–88. https://doi.org/10.54187/jnrs.1011739

[58]. **Liu, X., Wang, X., Tao, L., An, F., & Jiang, G.** (2023). Alleviating conditional independence assumption of naive Bayes. *Statistical Papers*, *65*(5), 2835–2863. https://doi.org/10.1007/s00362-023-01474-5

[59]. **Ye, C., Tsuchida, R., Petersson, L., & Barnes, N**. (2024). Label shift Estimation for Class-Imbalance Problem: A Bayesian approach. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1062–1071. https://doi.org/10.1109/wacv57701.2024.00111

[60]. **Budach, L., Feuerpfeil, M., Ihde, N., Nathansen, A., Noack, N., Patzlaff, H., Harmouch, H., & Naumann, F**. (2022). The effects of data quality on machine learning performance. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2207.14529

[61]. **Siang, L. C., Elnawawi, S., Rippon, L. D., O'Connor, D. L., & Gopaluni, R. B.** (2023). Data quality over Quantity: Pitfalls and Guidelines for process analytics. *IFAC-PapersOnLine*, *56*(2),7992–7999 https://doi.org/10.1016/j.ifacol.2023.10.921

[62]. **Berar, O.-A., Prošek, A., & Mavko, B.** (2013). RELAP5 and TRACE assessment of the Achilles natural reflood experiment. *Nuclear Engineering and Design, 261,* 306–316. https://doi.org/10.1016/j.nucengdes.2013.05.007

[63]. **Rodriguez, O.** (2013). RELAP5-3D thermal hydraulics computer program analysis coupled with DAKOTA and STAR-CCM+ codes

[64]. **Olatubosun, S. A., Ayodeji, A., & Amidu, M. A.** (2021). Safety assessment of AP1000: Common transients, analysis codes and research gaps. *Nuclear Engineering and Design, 375,* 111100. https://doi.org/10.1016/j.nucengdes.2021.111100

[65]. **Alberti, A. L., Agarwal, V., Gutowska, I., Palmer, C. J., & De Oliveira, C. R.** (2023). Automation levels for nuclear reactor operations: A revised perspective. *Progress in Nuclear Energy, 157,* 104559. https://doi.org/10.1016/j.pnucene.2022.104559

[66]. **Al-Ani, J.** (2021). *Development of a Nordic BWR plant model in APROS and design of a power controller using the control rods* (Master's thesis). KTH Royal Institute of Technology.

[67]. **Wang, P., Jiang, Q., Chen, Z., Liao, L., Xiao, K., & Wu, S.** (2021). Optimization of control rod travel for a small pressurized water reactor. *Progress in Nuclear Energy, 143,* 104050. https://doi.org/10.1016/j.pnucene.2021.104050

[68]. **Oludare, A., Agu, M., Umar, A., Adedayo, S., Omolara, O. E., & Okafor, L.** (2014). Assessing the Design effect of Pressure Vessel Height and Radius

on Reactor Stability and Safety. *Control Theory and Informatics*, *4*(1), 19–33. https://www.iiste.org/Journals/index.php/CTI/article/download/9404/9626

[69]. Zubair, M., Al Suwaidi, R. R., & Al Souqi, A. A. (2021). Behavior of emergency core cooling system (ECCS) during the early stage of loss of coolant accident (LOCA) for APR 1400 with Flownex software. *Progress in Nuclear Energy*, *141*, 103949. https://doi.org/10.1016/j.pnucene.2021.103949

[70]. Rossiter, G., & Peakman, A. (2023). Development and validation of Loss of Coolant Accident (LOCA) simulation capability in the ENIGMA fuel performance code for zirconium-based cladding materials. *Nuclear Engineering and Design*, *416*, 112767. https://doi.org/10.1016/j.nucengdes.2023.112767

[71]. Ming, Y., Shen, H., Cammi, A., Zhao, F., & Tian, R. (2024). Modeling and safety analysis of supercritical CO2 Brayton cycle reactor system under loss of coolant accident (LOCA). *Annals of Nuclear Energy*, *207*, 110740. https://doi.org/10.1016/j.anucene.2024.110740

[72]. Zha, D., Bhat, Z. P., Lai, K., Yang, F., & Hu, X. (2023). Data-centric AI: Perspectives and challenges. In *Society for Industrial and Applied Mathematics eBooks* (pp. 945–948). https://doi.org/10.1137/1.9781611977653.ch106

[73]. Henriksson, J. (2023). *Outlier detection as a safety measure for safety-critical deep learning* (Master's thesis). Chalmers University of Technology.

[74]. Althnian, A., AlSaeed, D., Al-Baity, H., Samha, A., Dris, A. B., Alzakari, N., Elwafa, A. A., & Kurdi, H. (2021). Impact of dataset size on classification performance: an empirical evaluation in the medical domain. *Applied Sciences*, *11*(2), 796. https://doi.org/10.3390/app11020796

[75]. Korjus, K., Hebart, M. N., & Vicente, R. (2016). An efficient data partitioning to improve classification performance while keeping parameters interpretable. *PLoSONE*, *11*(8),e0161788. https://doi.org/10.1371/journal.pone.0161788

[76]. Sadaiyandi, J., Arumugam, P., Sangaiah, A. K., & Zhang, C. (2023). Stratified Sampling-Based Deep Learning Approach to increase prediction

accuracy of unbalanced dataset. *Electronics*, *12*(21), 4423. https://doi.org/10.3390/electronics12214423

[77]. **Cabello-Solorzano, K., De Araujo, I. O., Peña, M., Correia, L., & Tallón-Ballesteros, A. J.** (2023). The impact of data normalization on the accuracy of machine learning algorithms: a comparative analysis. In *Lecture notes in networks and systems* (pp. 344–353). https://doi.org/10.1007/978-3-031-42536-3_33

[78]. **Sinsomboonthong, S.** (2022). Performance Comparison of New Adjusted Min-Max with Decimal Scaling and Statistical Column Normalization Methods for Artificial Neural Network Classification. *International Journal of Mathematics and Mathematical Sciences*, *2022*, 1–9. https://doi.org/10.1155/2022/3584406

[79]. **Santos, C. F. G. D., & Papa, J. P.** (2022). Avoiding Overfitting: A survey on regularization methods for convolutional neural networks. *ACM Computing Surveys*, *54*(10s), 1–25. https://doi.org/10.1145/3510413

[80]. **Mueller, J. P.** (2023). *Beginning programming with Python for dummies*. Wiley.

[81]. **Kaswan, K. S., Dhatterwal, J. S., & Balamurugan, B.** (2023). *Python for beginners*. CRC Press.

[82]. **Naik, P. G.** (2023). *Conceptualizing Python in Google Colab: Hands-on practical sessions*. ResearchGate.

[83]. **Mokhtarzadeh, H., & Bagheri, S.** (2023). Streamlining C3D File Processing and Visualization: A User-Friendly Approach Using Google Colab and Open-Source Python Packages. https://doi.org/10.31224/3088

APPENDINCES

Page

APPENDIX A

```
#Random Forest Block
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
x_train=pd.read_csv("/content/TrainingDataNormalizedX.csv")
y_train=pd.read_csv("/content/TrainingDataNormalizedY.csv")
x_validate=pd.read_csv("/content/ValidationDataNormalizedX.csv")

clf = RandomForestClassifier(n_estimators=100, criterion='gini', max_depth=80, min_samples_split=2,
clf.fit(x_train, y_train)

feature_importances = clf.feature_importances_
#sorted_indices = feature_importances.argsort()[::-1]
#sorted_feature_names = data.feature_names[sorted_indices]
##sorted_importances = feature_importances[sorted_indices]
clffeatureimportace=pd.DataFrame(feature_importances)
clffeatureimportace.to_csv("/content/clffeatureimportace.csv")
# Create a bar plot of the feature importances
#sns.set(rc={'figure.figsize':(11.7,8.27)})
#sns.barplot(sorted_importances, sorted_feature_names)
# Validate

a_tolist=clf.predict(x_validate)
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

```
#Logistic regression block
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.datasets import make_classification
x_train=pd.read_csv("/content/TrainingDataNormalizedX.csv")
y_train=pd.read_csv("/content/TrainingDataNormalizedY.csv")
x_validate=pd.read_csv("/content/ValidationDataNormalizedX.csv")
clf = LogisticRegression(penalty='elasticnet', dual=False, tol=0.0001, C=2000.0, fit_intercept=True, int
                         max_iter=100, multi_class='deprecated', verbose=0, warm_start=False, n_jobs=No
clffeatureimportace=pd.DataFrame(feature_importances)
clffeatureimportace.to_csv("/content/clffeatureimportace.csv")
a_tolist=clf.predict(x_validate)
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

Figure A.1: Codebase blocks for machine learning methods.

```
#Support Vector Block
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn import svm
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
x_train=pd.read_csv("/content/TrainingDataNormalizedX.csv")
y_train=pd.read_csv("/content/TrainingDataNormalizedY.csv")
x_validate=pd.read_csv("/content/ValidationDataNormalizedX.csv")
clf =svm.SVC()
clf.fit(x_train, y_train)
a_tolist=clf.predict(x_validate)
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

```
#Multilayer Perceptron
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import make_classification

x_train=pd.read_csv("/content/TrainingDataNormalizedX.csv")
y_train=pd.read_csv("/content/TrainingDataNormalizedY.csv")
x_validate=pd.read_csv("/content/ValidationDataNormalizedX.csv")

clf = MLPClassifier(hidden_layer_sizes=(200,2), activation='relu', solver='adam', alpha=0.0001,
clf.fit(x_train, y_train)
a_tolist=clf.predict(x_validate)
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
#NeighborsClassifier
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.neighbors import (NeighborhoodComponentsAnalysis,
KNeighborsClassifier)
from sklearn.pipeline import Pipeline
nca = NeighborhoodComponentsAnalysis(random_state=42)
knn = KNeighborsClassifier(n_neighbors=2)
nca_pipe = Pipeline([('nca', nca), ('knn', knn)])
x_train=pd.read_csv("/content/TrainingDataNormalizedX.csv")
y_train=pd.read_csv("/content/TrainingDataNormalizedY.csv")
x_validate=pd.read_csv("/content/ValidationDataNormalizedX.csv")
nca_pipe.fit(x_train, y_train)
a_tolist =nca_pipe.predict(x_validate)
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

Figure A.1: Codebase blocks for machine learning methods. (Continue)

```
#Decision Tree
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

# Import Files
dfx = pd.read_csv("/content/TrainingDataNormalizedX.csv")
dfy = pd.read_csv("/content/TrainingDataNormalizedY.csv")
dfvx = pd.read_csv("/content/ValidationDataNormalizedX.csv")
# Import necessary modules
from sklearn import tree

# Train
clf = tree.DecisionTreeClassifier(  criterion='gini', splitter='best', max_depth=None,
                                  min_weight_fraction_leaf=0.0, max_features='sqrt', r
clf = clf.fit(dfx , dfy)
feature_importances = clf.feature_importances_
#sorted_indices = feature_importances.argsort()[::-1]
#sorted_feature_names = data.feature_names[sorted_indices]
##sorted_importances = feature_importances[sorted_indices]
#clffeatureimportace=pd.DataFrame(feature_importances)
#clffeatureimportace.to_csv("/content/clffeatureimportace.csv")
# Create a bar plot of the feature importances
#sns.set(rc={'figure.figsize':(11.7,8.27)})
#sns.barplot(sorted_importances, sorted_feature_names)
# Validate
a_tolist = clf.predict(dfvx).tolist()
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

```
#Gradient Boosting
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.datasets import make_hastie_10_2
from sklearn.ensemble import GradientBoostingClassifier
dfx = pd.read_csv("/content/TrainingDataNormalizedX.csv")
dfy = pd.read_csv("/content/TrainingDataNormalizedY.csv")
dfvx = pd.read_csv("/content/ValidationDataNormalizedX.csv")
clf = GradientBoostingClassifier(loss='log_loss', learning_rate=0.2, n_estimators=100,
a_tolist = clf.predict(dfvx).tolist()
feature_importances = clf.feature_importances_
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
clffeatureimportace=pd.DataFrame(feature_importances)
clffeatureimportace.to_csv("/content/clffeatureimportace.csv")
```

```
#Naive Bayes
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.naive_bayes import GaussianNB
dfx = pd.read_csv("/content/TrainingDataNormalizedX.csv")
dfy = pd.read_csv("/content/TrainingDataNormalizedY.csv")
dfvx = pd.read_csv("/content/ValidationDataNormalizedX.csv")
clf = GaussianNB()
a_tolist =  clf.fit(dfx, dfy).predict(dfvx).tolist()
OUTPUT=pd.DataFrame(a_tolist)
OUTPUT.to_csv("/content/ValidationDataNormalizedY.csv")
```

Figure A.1: Codebase blocks for machine learning methods. (Continue)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import LearningCurveDisplay, learning_curve

# Import Files
dfx = pd.read_csv("/content/TrainingDataNormalizedX.csv")
dfy = pd.read_csv("/content/TrainingDataNormalizedY.csv")
dfvx = pd.read_csv("/content/ValidationDataNormalizedX.csv")
# Import necessary modules
from sklearn import tree

# Train
clf = tree.DecisionTreeClassifier(criterion='gini')
#clf = clf.fit(dfx , dfy)
train_size_abs, train_scores, test_scores = learning_curve(
    clf, dfx, dfy, train_sizes=[0.3, 0.6, 0.9])
display = LearningCurveDisplay(train_sizes=train_size_abs,
    train_scores=train_scores, test_scores=test_scores, score_name="Score")
display.plot()
plt.title("Decision Trees")
plt.show()
for train_size, cv_train_scores, cv_test_scores in zip(
    train_size_abs, train_scores, test_scores
):
    print(f"{train_size} samples were used to train the model")
    print(f"The average train accuracy is {cv_train_scores.mean():.2f}")
    print(f"The average test accuracy is {cv_test_scores.mean():.2f}")
    print(f"The average test variance is {cv_test_scores.var():.2f}")
```

Figure A.2: Codebase Blocks for Learning Curve Visualization.

APPENDIX B

Table B.1: Raw feature list for training and validation sets.

FEATURE	ELEMENT NAME	PARAMETER
mflowj508010000	Bypass Fuel Assembly (Hottest Channel)	Mass Flow
p508010000	Bypass Fuel Assembly (Hottest Channel)	Pressure
tempf508010000	Bypass Fuel Assembly (Hottest Channel)	Temperature
mflowj107010000	Coolant Pump 1	Mass Flow
p107010000	Coolant Pump 1	Pressure
tempf107010000	Coolant Pump 1	Temperature
mflowj207010000	Coolant Pump 2	Mass Flow
p207010000	Coolant Pump 2	Pressure
tempf207010000	Coolant Pump 2	Temperature
mflowj307010000	Coolant Pump 3	Mass Flow
p307010000	Coolant Pump 3	Pressure
tempf307010000	Coolant Pump 3	Temperature
mflowj407010000	Coolant Pump 4	Mass Flow
p407010000	Coolant Pump 4	Pressure
tempf407010000	Coolant Pump 4	Temperature
p505010000	DownComer Annulus	Pressure
tempf505010000	DownComer Annulus	Temperature
mflowj281000000	Feed Valve 1	Mass Flow
mflowj181000000	Feed Valve 2	Mass Flow
mflowj183000000	Feed Valve 3	Mass Flow
mflowj301010000	hot leg To Steam Generator 3	Mass Flow
p301010000	hot leg To Steam Generator 3	Pressure
tempf301010000	hot leg To Steam Generator 3	Temperature
mflowj401010000	hot leg To Steam Generator 4	Mass Flow
p401010000	hot leg To Steam Generator 4	Pressure
tempf401010000	hot leg To Steam Generator 4	Temperature
mflowj349000000	Loca Valve	Mass Flow
mflowj185000000	Main Steam Header Valve 1	Mass Flow
mflowj285000000	Main Steam Header Valve 2	Mass Flow

Table B.1: Raw feature list for training and validation sets. (Continue)

mflowj160000000	Reactor Inlet Header 1 to DownComer Annulus	Mass Flow
mflowj260000000	Reactor Inlet Header 2 to Downcomer	Mass Flow
mflowj308010000	Reactor Inlet Header 3	Mass Flow

p308010000	Reactor Inlet Header 3	Pressure
tempf308010000	Reactor Inlet Header 3	Temperature
mflowj408010000	Reactor Inlet Header 4	Mass Flow
p408010000	Reactor Inlet Header 4	Pressure
tempf408010000	Reactor Inlet Header 4	Temperature
sattemp100010000	Reactor Outlet Header 1	Volume Saturation Temperature Partial Pressure
tempf100010000	Reactor Outlet Header 1	Temperature
tsatt100010000	Reactor Outlet Header 1	Saturation Temperature Total pressure
voidg100010000	Reactor Outlet Header 1	Volume vapor fraction
mflowj300010000	Reactor Outlet Header 3	Mass Flow
p300010000	Reactor Outlet Header 3	Pressure
tempf300010000	Reactor Outlet Header 3	Temperature
mflowj400010000	Reactor Outlet Header 4	Mass Flow
p400010000	Reactor Outlet Header 4	Pressure
tempf400010000	Reactor Outlet Header 4	Temperature
rktpow0	Reactor Power	Reactor Power
mflowj528000000	Relief Valve 1	Mass Flow
mflowj529000000	Relief Valve 2	Mass Flow
mflowj530000000	Relief Valve 3	Mass Flow
mflowj302010000	Steam Generator Lower 3	Mass Flow
p302010000	Steam Generator Lower 3	Pressure
tempf302010000	Steam Generator Lower 3	Temperature
mflowj402010000	Steam Generator Lower 4	Mass Flow
p402010000	Steam Generator Lower 4	Pressure
tempf402010000	Steam Generator Lower 4	Temperature
mflowj303010000	Steam Generator Medium 3	Mass Flow
p303010000	Steam Generator Medium 3	Pressure
tempf303010000	Steam Generator Medium 3	Temperature
mflowj403010000	Steam Generator Medium 4	Mass Flow

Table B.1: Raw feature list for training and validation sets. (Continue)

p403010000	Steam Generator Medium 4	Pressure
tempf403010000	Steam Generator Medium 4	Temperature
mflowj306010000	Steam Generator Out to Main Pump 3	Mass Flow
p306010000	Steam Generator Out to Main Pump 3	Pressure

tempf306010000	Steam Generator Out to Main Pump 3	Temperature
mflowj406010000	Steam Generator Out to Main Pump 4	Mass Flow
p406010000	Steam Generator Out to Main Pump 4	Pressure
tempf406010000	Steam Generator Out to Main Pump 4	Temperature
mflowj305010000	Steam Generator Outlet/Joint Cold Leg 3	Mass Flow
p305010000	Steam Generator Outlet/Joint Cold Leg 3	Pressure
tempf305010000	Steam Generator Outlet/Joint Cold Leg 3	Temperature
mflowj405010000	Steam Generator Outlet/Joint Cold Leg 4	Mass Flow
p405010000	Steam Generator Outlet/Joint Cold Leg 4	Pressure
tempf405010000	Steam Generator Outlet/Joint Cold Leg 4	Temperature
p134010000	Steam Generator Secondary 1	Pressure
p234010000	Steam Generator Secondary 2	Pressure
mflowj304010000	Steam Generator Upper 3	Mass Flow
p304010000	Steam Generator Upper 3	Pressure
tempf304010000	Steam Generator Upper 3	Temperature
mflowj404010000	Steam Generator Upper 4	Mass Flow
p404010000	Steam Generator Upper 4	Pressure
tempf404010000	Steam Generator Upper 4	Temperature
mflowj468000000	Turbine Stop Valve	Mass Flow
p510010000	Upper Plenum 1	Pressure
mflowj151000000	Upper Plenum To Reactor Outlet Header 1	Mass Flow
cntrlvar1	Water Level Steam Generator 1	Water Level
cntrlvar2	Water Level Steam Generator 2	Water Level
cntrlvar3	Water Level Steam Generator 3	Water Level
cntrlvar4	Water Level Steam Generator 4	Water Level
cntrlvar5	Water Level Pressurizer	Water Level

CURRICULUM VITAE

Name Surname **:** Ceyhun Yavuz

EDUCATION **:**

- **B.Sc.** **:** 2008, ITU, Physics Engineering, Physics Engineering Department
- **M.Sc.** **:** 2015, ITU, Faculty, Department**,** Energy Science and Technology **,** Energy Institute

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Yavuz, C., Şentürk Lüle, S.** 2022. The Application of Artificial Intelligence to Nuclear Power Plant Safety. In: Mercier-Laurent, E., Kayakutlu, G. (eds) *Artificial Intelligence for Knowledge Management, Energy, and Sustainability. AI4KMES 2021. IFIP Advances in Information and Communication Technology*, vol 637. Springer, Cham. https://doi.org/10.1007/978-3-030-96592-1_9
- **Yavuz, C., Şentürk Lüle, S.** Pressurized Water Reactor Transient Detection With Artificial Intelligence to Support Reactor Operators, *Science and Technology of Nuclear Installations*, 2025, 1443278, 12 pages, 2025. https://doi.org/10.1155/stni/1443278

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Ozgener, H. A.,Yavuz, C**., 2019. An Analytical Solution for the Two-Group Diffusion Theory Analysis of Multiregion Source-Driven Systems. *ASME. ASME J of Nuclear Rad Sci*. July 2019; 5(3): 031401. https://doi.org/10.1115/1.4042021