A

Major Project

On

ADVANCE DETECTION OF MACHINE FAILURE IN AUTOMATED INDUSTRIES USING ML ALGORITHMS

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

BY

Yamini Seepana	(177R1A05A6)
Priya Varshini Chapaala	(177R1A0566)
Kallam Raja Reddy	(177R1A0590)

Under the Guidance of

Dr. K. SRUJAN RAJU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)Recognized Under Section 2(f) & 12(B) of the UGC Act.1956,
Kandlakoya (V), Medchal Road, Hyderabad-501401.

2017-2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**ADVANCE DETECTION OF MACHINE FAILURE IN AUTOMATED INDUSTRIES USING ML ALGORITHMS**" being submitted by **YAMINI SEEPANA (177R1A05A6), PRIYA VARSHINI CHAPAALA (177R1A0566) & KALLAM RAJA REDDY (177R1A0590)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering of the Jawaharlal Nehru Technological University, Hyderabad, during the year 2021. It is certified that they have completed the project satisfactorily.

INTERNAL GUIDE
Dr. K. Srujan Raju

DIRECTOR
Dr. A. Raji Reddy

HOD
Dr. K. Srujan Raju

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to my guide.

Dr. K. Srujan Raju Professor, CSE department for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators: **Mr. J. Narasimha Rao, Dr. B. Krishna, Dr. Suwarna Gothane, Mr. K. Murali,** and **Ms. Najeema Afrin** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to the Head of the Department **Dr. K. Srujan Raju** for providing excellent infrastructure and a nice atmosphere for completing this project successfully.

We are obliged to our Director **Dr. A. Raji Reddy** for being cooperative throughout the course of this project. We would like to express our sincere gratitude to our Chairman Sri. **Ch. Gopal Reddy** for his encouragement throughout the course of this project.

The guidance and support received from all the members of **CMR TECHNICAL CAMPUS** who contributed and who are contributing to this project, was vital for the success of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity thank our family for their constant encouragement without which this assignment would not be possible. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project.

YAMINI SEEPANA (177R1A05A6)

PRIYA VARSHINI CHAPAALA (177R1A0566)

KALLAM RAJA REDDY (177R1A0590)

ABSTRACT

Performance monitoring and failure prediction of industrial equipment plays a very important role not only in the quality of the manufactured material but also in the amount of time and money saved in the overall maintenance. Fault prediction before testing provides feedback for procedures of maintenance, resulting in cost savings.

As part of this Major Project, we try to build a Software System that can predict the failure of machinery based on the enormous amount of data sets available from the plants. We will be using a model based on CNN-LSTM, to predict faults stages of the equipment at an early stage and use a classifier to alert the system. LSTM algorithm is used for creating models and this model is trained using the data set of equipment. This trained model is then used to monitor the Machinery and predict the faults ahead of time and alert the user.

LIST OF FIGURES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1.INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

If maintenance of machines is not done at appropriate time intervals the cost of failure will be much higher than its apparent cost. Currently, most companies deal with this problem by being pessimistic and through precise maintenance programs to replace fallible components before failures. Although regular maintenance is better than failures, we often end up doing the maintenance before it's needed. Hence, it is not an optimal solution from a cost perspective.

Predictive Maintenance using Machine Learning Algorithms can be one such optimal solution.

1.2 PROJECT PURPOSE

This has been developed to facilitate the identification, retrieval of the items and information. System is built with manually exclusive features. In all cases system will specify object which are physical or on performance characteristics. They are used to give optimal distraction and other information. Data are used for identifying, accessing, storing and matching records. The data ensures that only one value of the code with a single meaning is correctly applied to give entity or attribute as described in various ways.

1.3 PROJECT FEATURES

Data from NASA TURBOFAN ENGINE has been put together to build this dataset that takes three datasets as inputs which include training data, testing data and ground truth data. The variables comprise of different factors such as 100 unique engine Id's ,cycles of each engine,sensor measurement and operational settings.There are 26 feature columns in the dataset and all of them play an important in determining the failure of engine.

2.SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

2.2 EXISTING SYSTEM

Traditionally maintenance has been done using SCADA systems(computer system for gathering and analysing real time data) set up with human coded thresholds, alert rules and configurations.

Three main directions followed in the previous studies on IOT based applications are:

1. Defining and specifying metrics to calculate complexity of software
2. Verifying correctness and validating thoroughness

Some metrics have been presented to describe software quality in forms of static and dynamic platforms. In the static platform, features of code structure are measured as metrics.Static measurements can be a number of supervisors.

Dynamic platforms measure testing perfectionism. Basic element measurements depend on auxiliary and information stream scope. The connection between product measurements and blame inclination, and also numerous quantifiable programming characteristics has been exactly demonstrated by many researchers.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

This semi-manual approach doesn't take into account the more complex dynamic behavioral patterns of the machinery, or the contextual data relating to the manufacturing process at large.

Moreover, the threshold values which are provided manually may not be valid at all times.

2.3 PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The existing system has several disadvantages and many more difficulties to work with. The proposed system tries to eliminate or reduce these difficulties up to some extent.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

Using ML for Predictive Maintenance eliminates most of the guesswork.

ML enables to:

- Create predictive models for maximizing asset lifetime, operational efficiency, or uptime.
- Leverage past and continuous data
- Optimize the periodic maintenance operations.
- Avoid or minimize the down times.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company.

Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it give an indication of the system is economically possible for development.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

1. CPU: 2 x 64-bit 2.8 GHz 8.00 GT/s CPUs.
2. RAM: 32 GB (or 16 GB of 1600 MHz DDR3 RAM)
3. Storage: 300 GB

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements-

Operating System supported:

1. Windows 7 or later
2. MacOS
3. Linux

Technologies and Languages used to Develop:

1. Python

Environment used:

1. Anaconda Jupyter Notebook

3.ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCITECTURE

This project architecture describes about how a data will be stored in database. This explains how the entire process is carried out .The detailed architecture is explained below.

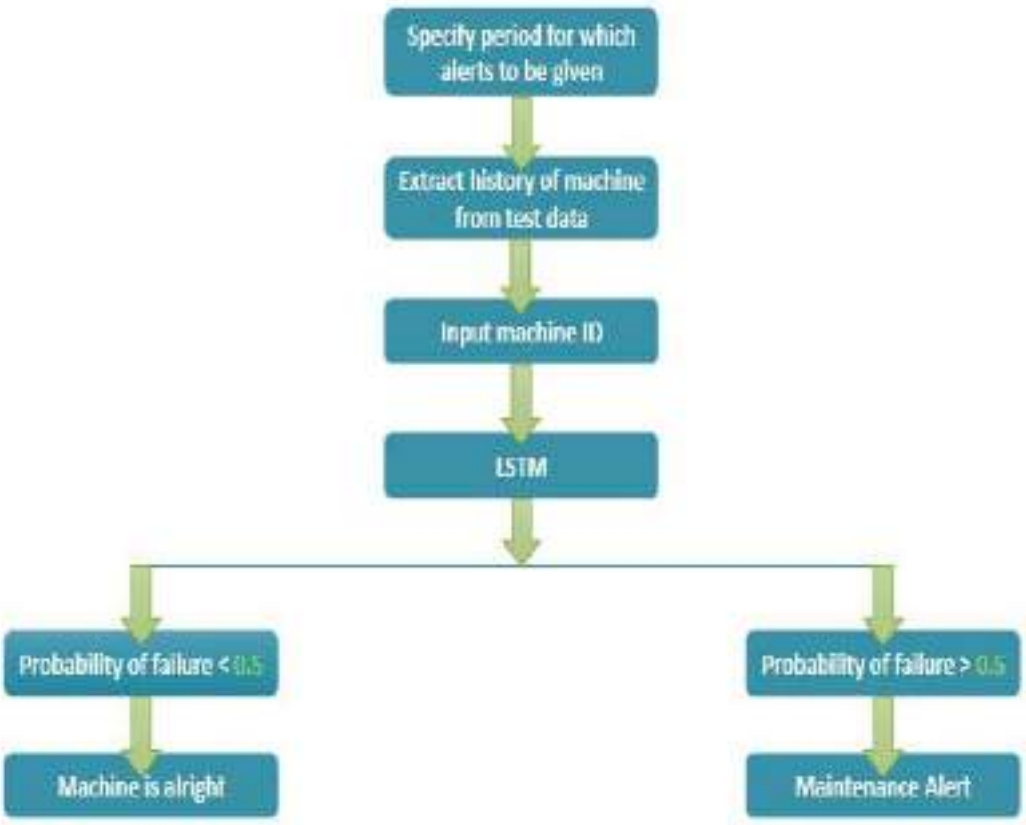


Fig. 3.1.1: Project Architecture

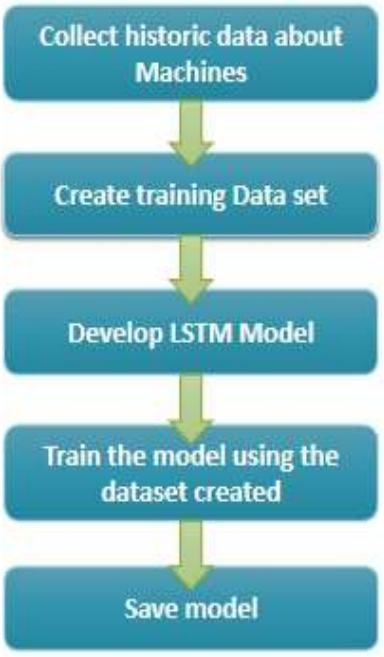


Fig. 3.1.2: Training Architecture

3.2 MODULES DESCRIPTION

Modules

- Dataset
- Jupiter
- Pandas
- Sklearn
- Keras

3.2.1 DATASET

DATA REQUIREMENTS:

As with any machine learning project, data is a key point. It is important to set aside a proper amount of unseen data for testing purposes.This will make it possible to evaluate the predictive models' quality before shipping them to a production setting.

DATA GATHERING:

The first question to answer is what kind of data should be gathered?
Gather everything you can. Always try to collect everything that seems remotely relevant. As it is often said, "Data is the new oil".
While collecting data some of the crucial questions which have to be answered are

- Where is predictive maintenance possible?
- What component should we focus on first?
- How does this component behavior vary across different locations, weather conditions, or workloads?
- Who are our domain expert partners? Their knowledge and experience will be considered to determine what kind of data to collect.

AMOUNT OF DATA:

The amount of data needed varies a lot from case to case, but in general, the most important part is to have a fair amount of failure examples. Usually, malfunctions are rare, and everything will need to be stored until enough examples are recorded.

Continuous monitoring of sensors will generate a lot of data that can be hard to manage, but big data solutions exist for a reason. It is essential to define how often the measurement from a sensor will be read, where the data will be stored, and how to process the values obtained. Storing time series data is a well-known problem, and there are plenty

of cost-efficient solutions that take advantage of the fact that the data is never updated and rarely queried.

DATASET USED:

The dataset we used for this python project is NASA TURBOFAN JET ENGINE DATASET

This project takes three datasets as inputs.

Training data: It is the aircraft engine run-to-failure data.

Testing data: It is the aircraft engine operating data without failure events recorded.

Ground truth data: It contains the information of true remaining cycles for each engine in the testing data.

DATA SCHEMA:

There are 26 feature columns in the dataset and all of them play an important in determining the failure of engine. Let us see the features and their data types.The data schema for the training and testing data is shown in the following table.

Index	Data fields	Type	Descriptions
1	id	Integer	aircraft engine identifier, range [1, 100]
2	cycle	Integer	time, in cycles
3	setting1	Double	operational setting 1
4	setting2	Double	operational setting 2
5	setting3	Double	operational setting 3
6	s1	Double	sensor measurement 1
7	s2	Double	sensor measurement 2
...	...		
26	s21	Double	sensor measurement 21

Fig. 3.2.1: Data schema

The input data consists of "PM_train.txt", "PM_test.txt", and "PM_truth.txt" in the orginal data source.

The training data ("PM_train.txt ") consists of multiple multivariate time series with "cycle" as the time unit , together with 21 sensor readings for each cycle. Each time series can be assumed as being generated from a different engine of the same type. Each engine is assumed to start with different degrees of initial wear and manufacturing variation and this information is unknown to the user. In this simulated data, the engine is assumed to be operating normally at the start of each time series. It starts to degrade at some point during the series of the operating cycles.

The degradation progresses and grows in magnitude.When a predefined threshold is reached, then the engine is considered unsafe for further operation. In other words, the last cycle in each time series can be considered as the failure point of the corresponding engine. Taking the sample training data shown in the following table as an example, the engine with id=1 fails at cycle 192, and engine with id=2 fails at cycle 287.

Sample training data

~20k rows,
100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21
1	1	-0.0007	-0.0004	100	518.67	641.82	1589.7		100	39.06	23.419
1	2	0.0019	-0.0003	100	518.67	642.15	1591.82		100	39	23.4236
1	3	-0.0043	0.0003	100	518.67	642.35	1587.99		100	38.95	23.3442
...	...										
1	191	0	-0.0004	100	518.67	643.34	1602.36		100	38.45	23.1295
1	192	0.0009	0	100	518.67	643.54	1601.41		100	38.48	22.9649
2	1	-0.0018	0.0006	100	518.67	641.89	1583.84		100	38.94	23.4585
2	2	0.0043	-0.0003	100	518.67	641.82	1587.05		100	39.06	23.4085
2	3	0.0018	0.0003	100	518.67	641.55	1588.32		100	39.11	23.425
...	...										
2	286	-0.001	-0.0003	100	518.67	643.44	1603.63		100	38.33	23.0169
2	287	-0.0005	0.0006	100	518.67	643.85	1608.5		100	38.43	23.0848

Sample testing data

~13k rows,
100 unique engine id

id	cycle	setting1	setting2	setting3	s1	s2	s3	...	s19	s20	s21
1	1	0.0023	0.0003	100	518.67	643.02	1585.29		100	38.86	23.3735
1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45		100	39.02	23.3916
1	3	0.0003	0.0001	100	518.67	642.46	1586.94		100	39.08	23.4166
...	...										
1	30	-0.0025	0.0004	100	518.67	642.79	1585.72		100	39.09	23.4069
1	31	-0.0006	0.0004	100	518.67	642.58	1581.22		100	38.81	23.3552
2	1	-0.0009	0.0004	100	518.67	642.66	1589.3		100	39	23.3923
2	2	-0.0011	0.0002	100	518.67	642.51	1588.43		100	38.84	23.2902
2	3	0.0002	0.0003	100	518.67	642.58	1595.6		100	39.02	23.4064
...	...										
2	48	0.0011	-0.0001	100	518.67	642.64	1587.71		100	38.99	23.2918
2	49	0.0018	-0.0001	100	518.67	642.55	1586.59		100	38.81	23.2618
3	1	-0.0001	0.0001	100	518.67	642.03	1589.92		100	38.99	23.296
3	2	0.0039	-0.0003	100	518.67	642.23	1597.31		100	38.84	23.3191
3	3	0.0006	0.0003	100	518.67	642.98	1586.77		100	38.69	23.3774
...	...										
3	125	0.0014	0.0002	100	518.67	643.24	1588.64		100	38.56	23.227
3	126	-0.0016	0.0004	100	518.67	642.88	1589.75		100	38.93	23.274

Sample ground truth data

100 rows

RUL
112
98
69
82
91

Fig. 3.2.2: Datasets

3.2.2 JUPITER

The Jupyter Notebook is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupiter.

3.2.3 PANDAS

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

3.2.4 SKLEARN

Scikit-learn package of python is later used to run different machine learning regression models on the data to perform analysis and make predictions on the data. The model’s accuracy and testing is done using the metrics package of the same package as well.

3.2.5 KERAS

Keras is an open-source software library that provides a Python interface for artificial neural networks.Keras is an open-source software library that provides a Python interface for artificial neural networks.

3.3 DATA PREPROCESSING

1. Column named ‘id’ which stores index+1 value is added to the pm_truth table.

```
In [3]: pm_truth=pd.read_csv('PM_truth.txt',sep=' ',header=None).drop([1],axis=1)
        pm_truth.columns=['more']
        pm_truth['id']=pm_truth.index+1
        pm_truth.head()
```

Out[3]:

	more	id
0	112	1
1	98	2
2	69	3
3	82	4
4	91	5

2. Calculate maximum number of cycles run by each machine by grouping data using attribute id.

```
In [5]: # generate column max for test data
        rul = pd.DataFrame(dataset_test.groupby('id')['cycle'].max()).reset_index()
        rul.columns = ['id', 'max']
        rul.head()
```

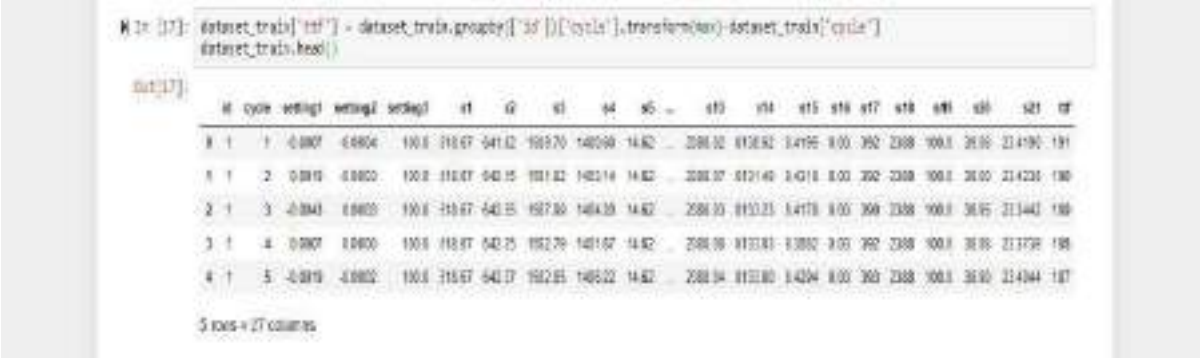
Out[5]:

	id	max
0	1	31
1	2	49
2	3	126
3	4	106
4	5	98

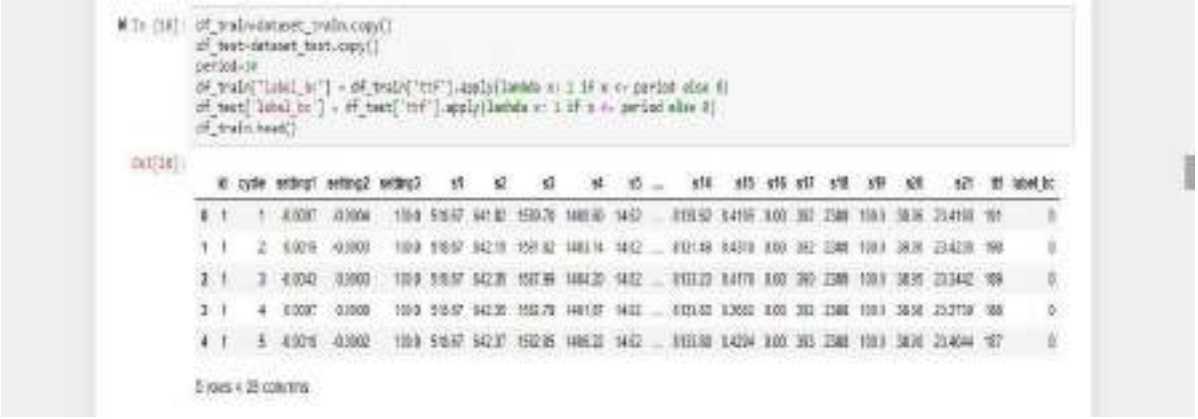
3. Find run to failure by adding column ‘more’ from PM_truth and column ‘max’ from rul.

4. Calculate time to failure for each row of testing data by subtracting cycle from rtf.

5. Calculate Time To Failure for each row of training data by subtracting cycle from rtf.



6. Add a new column label_bc which takes values 0 if ttf<period or 1 if ttf>period.



3.4 UML DIAGRAMS

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

3.4.1 USE CASE DIAGRAM

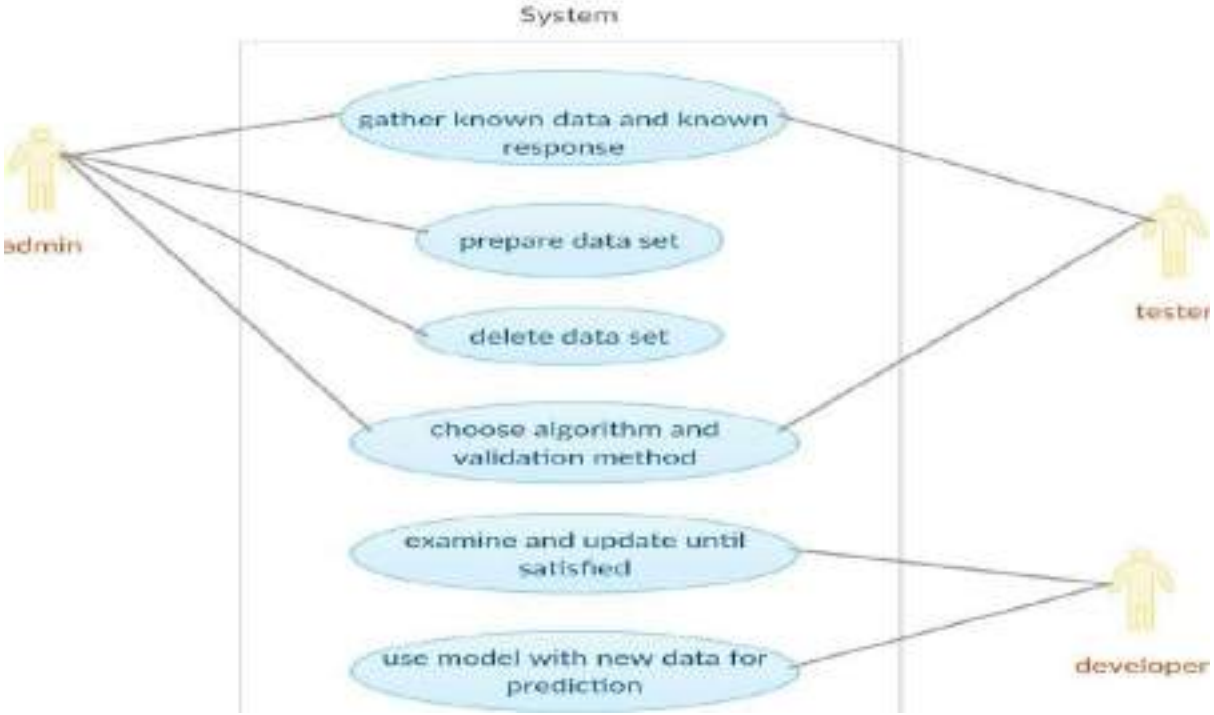


Fig 3.4.1: Use case diagram for admin,tester and developer

3.4.2 SEQUENCE DIAGRAM

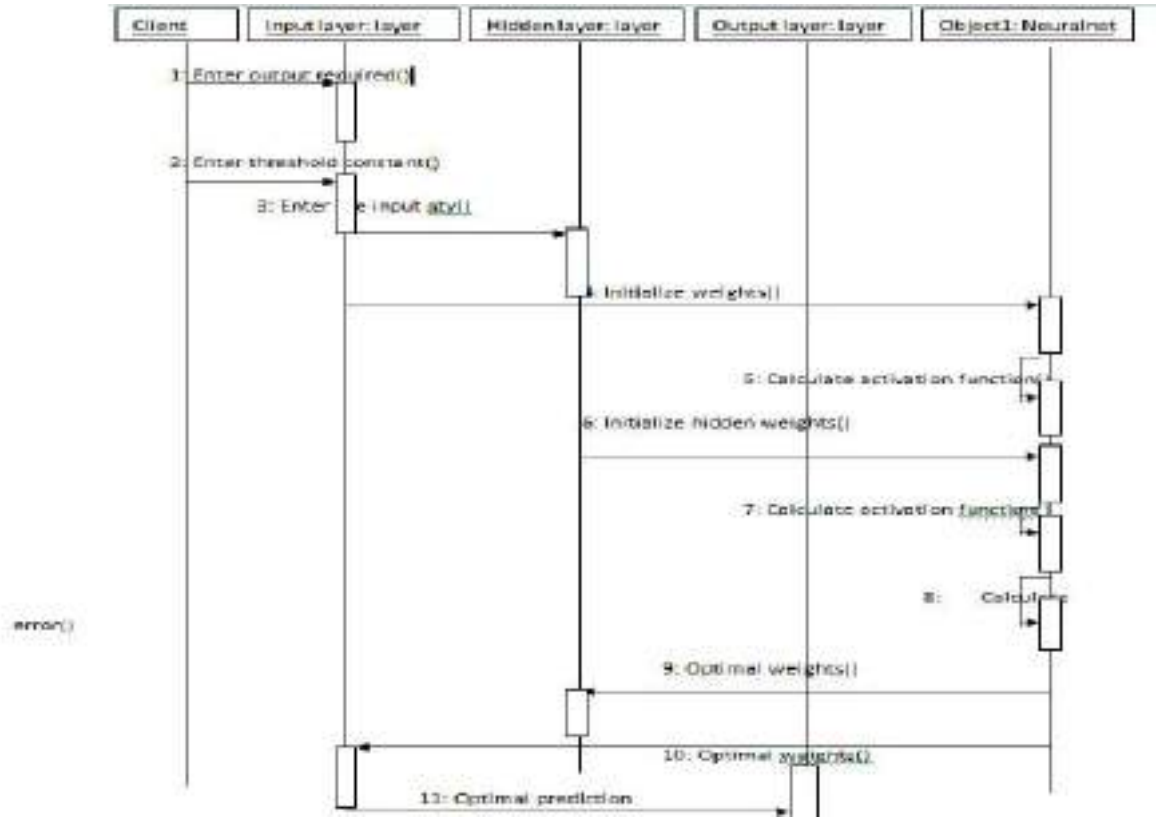


Fig 3.4.2: Sequence Diagram for Advance Detection of Machine Failure

3.4.3 ACTIVITY DIAGRAM

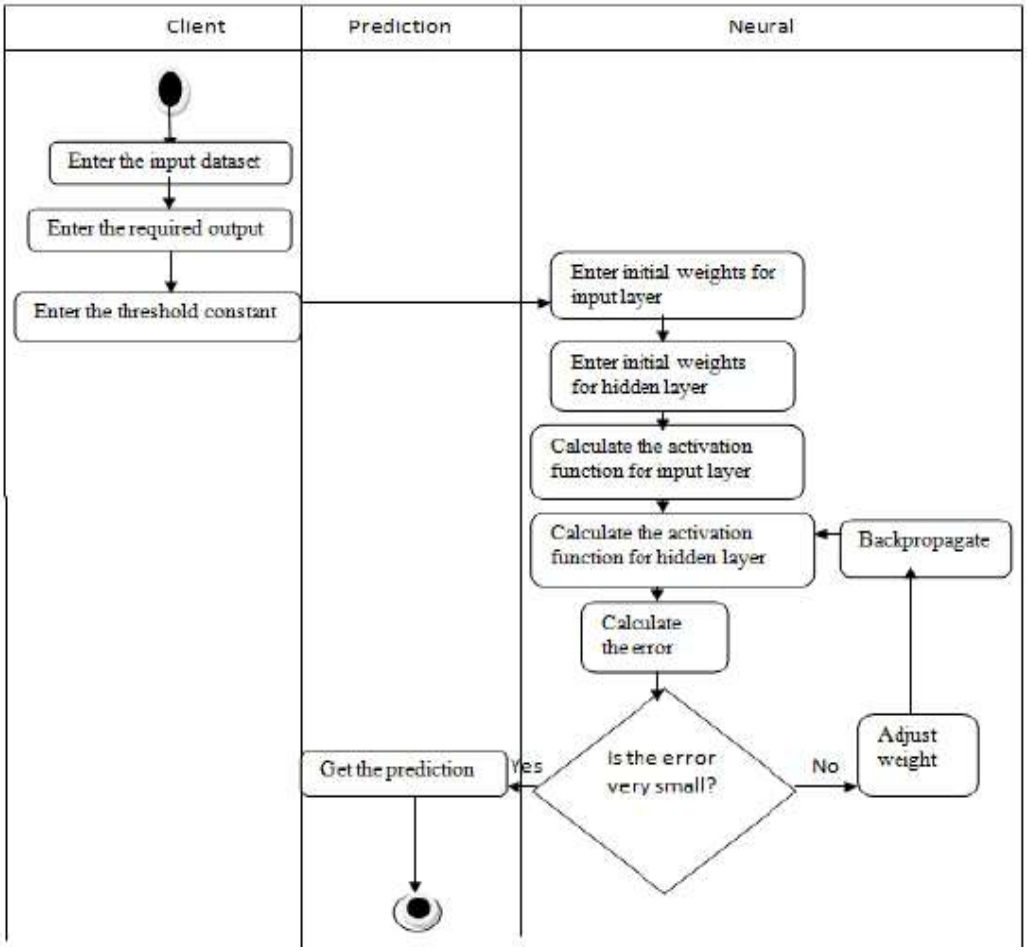


Fig 3.4.3: Activity Diagram for Advance Detection of Machine Failure

3.5 OVERVIEW

The quality of Internet of Things (IoT) applications has grown rapidly throughout recent years, and issues related to that have gained more importance for software developers .A serious step for changing the testing procedure is based on the capacity of assessing the fault prediction process, evaluating the degree of a product module and then testing.

Fault prediction before testing helps the developer to eliminate costs and fault detection after testing provides feedback for procedures of maintenance. Programming deficiencies can be straightforwardly measured at the programming phase. Nonetheless, finding relations between quantifiable software properties and faults can help detecting faults. Traditional methods, testing or simulation, cannot overcome the challenges posed in fault prediction. Two important points in this regard are high cost and time overhead.

In addition, re-enactment is not suitable for supporting transient properties since all the possible states of the framework doesn't considered. To solve this problem formal methods can be employed. Formal methods are based on mathematical logic. Generally, formal methods can be divided into formal specification and formal verification. Interaction behaviors between fault proneness and formal verification are called formal specifications. Most of the papers on fault prediction examine their proposed method through simulation and experiments. Another way of verifying an information-centric IoT application is model checking.

With the continuous development of machine learning, the concept of deep learning has been proposed. Deep learning is to approximate complex nonlinear functions with small errors through multi-layer information processing and feature extraction. Convolutional neural networks and recurrent neural networks, which have been proposed in the literature, have practical significance in the field of deep learning. Recurrent neural networks have higher accuracy in time series prediction. But deep training can cause the gradient of the neural network to disappear.

The long short-time memory network proposed adds three 'gate' structures on the basis of the recurrent neural network and has a new improvement in the prediction of time series, with relatively high accuracy. The output prediction of power plant equipment often has a large number of multi-dimensional influence factors. Although the convolutional neural network can usually process multi-dimensional data, the timing of the data cannot be well reflected. The introduction of traditional particle swarms, attention mechanisms and other algorithms can process multidimensional data, reducing data redundancy and data loss, but these methods often ignore the relationship between input data. In order to find an effective solution, this paper proposes a model based on LSTM applied to the fault prediction stage of the equipment. The extracted feature vector is put into the long short-term memory network, which is good at predicting time series data. This model not only ensures the characteristics of the input data and its relationship, but also ensures the timing of the model. Finally, the model prediction value, LSTM network prediction value and practical value are compared.

The rationality and effectiveness of the method are verified by an example. It is also proved that the method has a higher improvement effect than LSTM network prediction.

3.6 TYPES OF MAINTENANCE

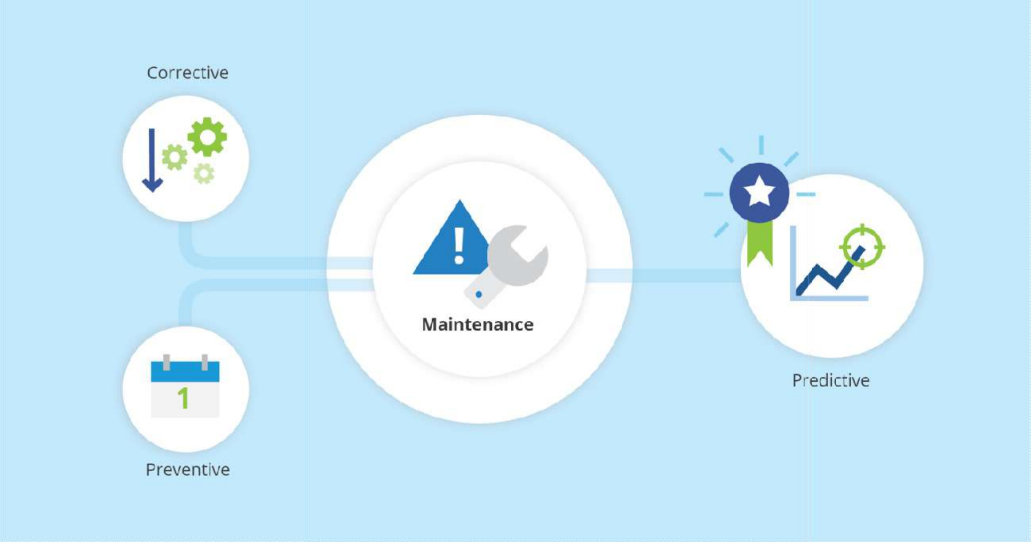


Fig 3.6: Types of Maintenance

3.6.1 CORRECTIVE MAINTENANCE

Most businesses rely on corrective maintenance (i.e., reactive), where the failing parts are replaced once they stop being functional to the system. This ensures parts are used entirely and, therefore, it doesn't waste a component's useful lifetime. This option adds the cost of down times, labor, and unscheduled maintenance.

It is the most straight-forward maintenance strategy, but the most expensive one, since downtime and unplanned maintenance costs can strongly affect productivity and returns.

While going on highway if your car breaks, you will call a mechanic, wait for him to arrive, then evaluate and fix the problem or take it to the mechanic shop. You got most of the equipment which failed, but lots of money and time were spent after breakage.

3.6.2 PREVENTIVE MAINTENANCE

A second, more robust approach is preventive maintenance (i.e., periodic), where components are replaced after a given time, regardless of their condition. This approach avoids catastrophic failures and unscheduled down times but requires careful consideration when determining a part's useful lifespan and replacements' periodicity.

It is generally a practical approach to avoid failures. However, unnecessary corrective actions can be taken. Sometimes replacing components that could still last longer leads to an increase in the operative costs.

In the above mentioned scenario, you will take the car to the mechanic shop periodically for service to maintain the car condition.

3.6.3 PREDICTIVE MAINTENANCE

Finally, predictive maintenance aims to optimize the balance between corrective and preventive maintenance by enabling just in time replacement of components. This approach minimizes the cost of unscheduled maintenance and maximizes the component's lifespan, thus getting more value out of a part. It is based on continuous monitoring of a machine or process integrity, allowing maintenance to be performed only when necessary.

Moreover, it allows the early detection of failures thanks to predictive tools based on historical data with machine learning techniques, integrity factors as analyzing visual aspects like wear or coloration, statistical inference methods, and other engineering approaches.For the car example, this will be the case when the car's computer indicates that it is time to make a specific revision.

KEY BENEFITS OF PREDICTIVE MAINTENANCE

Predictive maintenance is not the easiest solution to implement, but it's benefits are outstanding. If implemented well, these solutions will result in significant cost savings, mainly by maximizing the components' lifespan.

Replacements or machinery service will only take place when it is absolutely necessary, which will unload part of your maintenance team to focus on more exciting tasks (make sure to keep them around in case of emergency, though).

Since this approach measures components' real behavior, it can anticipate failures even in faulty pieces that will not last as long as we expected, something that preventive maintenance wouldn't do.

FROM SIMPLE AUTOMATION TO MACHINE LEARNING

Rule-based systems are a good starting point. Still, this approach will quickly lead to making adjustments and maintenance of hundreds of rules. If each rule has to be adapted for every machine, we have to adjust these values through time, as the environmental conditions can change, and the thresholds defined for today may be invalid in a couple of months. A simple example would be a seasonal change of temperature that can impact the machinery's normal operation values.

In this scenario, machine learning can make your life easier by automatically finding the hidden patterns in your data

3.7 LONG TERM SHORT MEMORY NEURAL NETWORK

WHAT ARE NEURAL NETWORKS?

Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

Neural networks reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning.

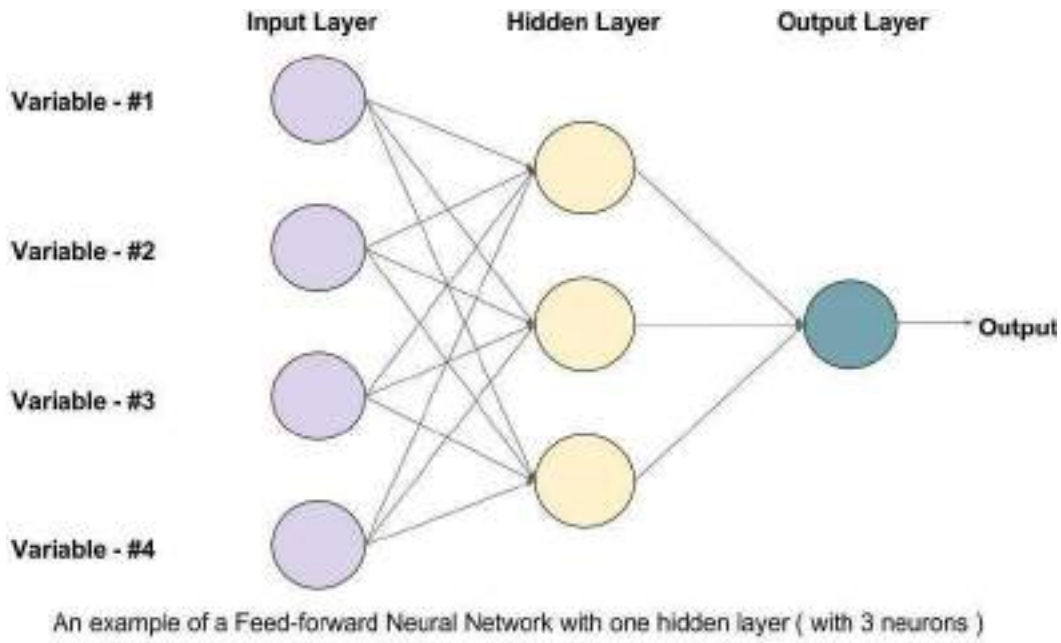


Fig. 3.7: Neural Network

WHAT DO THEY DO?

Neural networks are sometimes called Artificial Neural networks(ANN's), because they are not natural like neurons in your brain. They artificially mimic the nature and functioning of Neural networks. ANN's are composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.

Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on.

WHY NOT NEURAL NETWORKS/FEED FORWARD NETWORKS?

The Neural Network functions like a black box. You feed in some inputs from one side, you receive some outputs from the other side. The decision it makes is mostly based on the current inputs.

It's unfair to say that the neural network has no memory at all. After all, those learnt weights are some kind of memory of the training data. But this memory is more static. Sometimes we want to remember an input for later use.

The naive way is to let neural networks accept time series data by connecting several neural networks together. Each of the neural networks handles one time step. Instead of feeding the data at each individual time step, you provide data at all time steps within a window, or a context, to the neural network.

To overcome the challenge of understanding previous output we use Recurrent Neural Network.

WHAT ARE RNN's?

The idea behind RNNs is to make use of sequential information. In a traditional neural network we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations and you already know that they have a "memory" which captures information about what has been calculated so far.

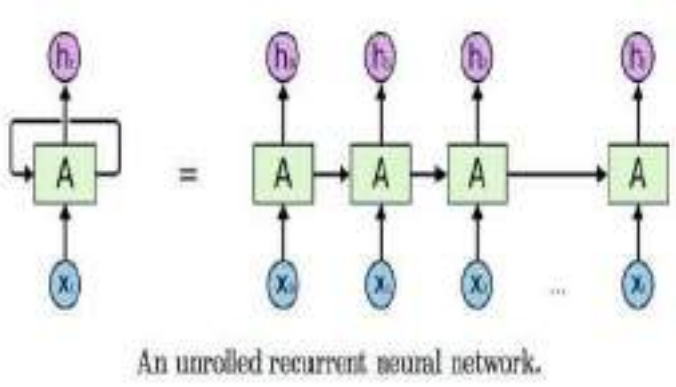


Fig. 3.7.1: RNN

RNN's suffer from two problems:vanishing gradient and exploding gradient, which make it unusable.

LSTM (long short term memory) was invented to solve this issue by explicitly introducing a memory unit, called the cell into the network.

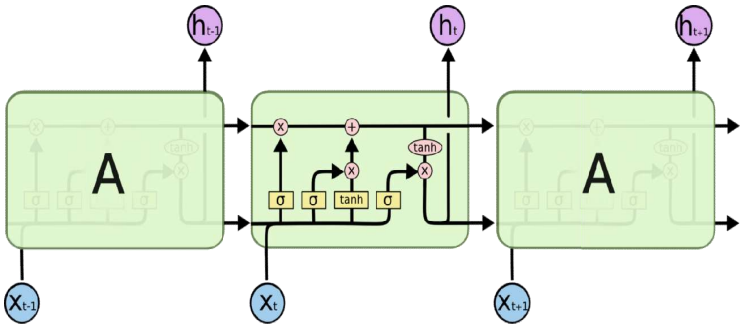


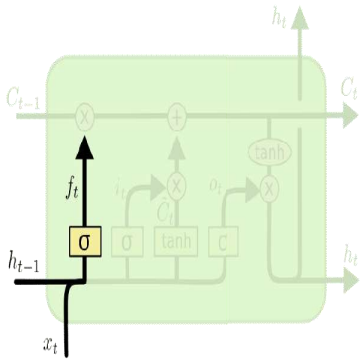
Fig. 3.7.2: LSTM

Every LSTM module will have 3 gates named as Forget gate, Input gate, Output gate.

FORGET GATE:

Decides how much o f the past you should remember.

This gate decides which information to be omitted in from the cell in that particular time stamp. It is decided by the sigmoid function. It looks at the previous state(ht-1) and the content input(Xt) and outputs a number between 0(omit this) and 1(keep this)for each number in the cell state Ct−1.



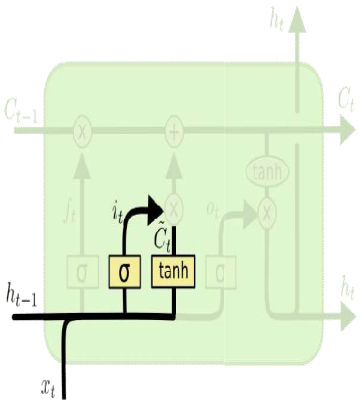
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Fig. 3.7.3: Forget Gate

UPDATE GATE/INPUT GATE:

Decides how much of this unit is added to the current state.

Sigmoid function decides which values to let through 0, 1 and tanh function gives weightage to the values which are passed deciding their level of importance ranging from -1 to 1**.**



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$$

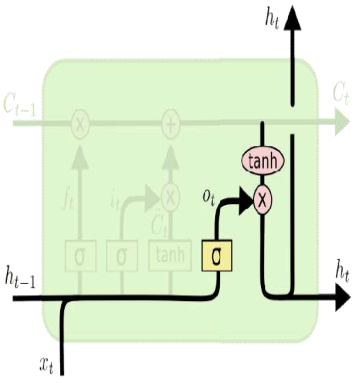
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Fig. 3.7.4: Update Gate/Input Gate

OUTPUT GATE:

Decides which part of the current cell makes it to the output.

Sigmoid function decides which values to let through 0,1 and tanh function gives weightage to the values which are passed deciding their level of importance ranging from -1 to 1 and multiplied with output of Sigmoid.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Fig. 3.7.5: Output Gate

STEPS FOR PREDICTING FAILURE OF A MACHINE

This project uses the example of simulated aircraft engine run-to-failure events to demonstrate the predictive maintenance modeling process. The implicit assumption of modeling data as done below is that the asset of interest has a progressing degradation pattern, which is reflected in the asset's sensor measurements. By examining the asset's sensor values over time, the machine learning algorithm can learn the relationship between the sensor values and changes in sensor values to the historical failures in order to predict failures in the future.

STEPS:

- Data pre-processing
- Feature scaling
- Preparing data for LSTM
- Training and evaluating

4.IMPLEMENTATION

4. IMPLEMENTATION

4.1 SAMPLE CODE


```
{
"cells": [
{
"cell_type": "markdown",
"metadata": {
"_uuid": "301a3e54e6e35c277a77a9098390f18b6c536c2c"
},
"source": [
"# ADVANCE PREDICTION OF MACHINE FAILURE"
]
},
{
"cell_type": "code",
"execution_count": 1,
"metadata": {
"_uuid": "f7b8ba8ca53b67ae95c36e2b37666a5613578983"
},
"outputs": [],
"source": [
"import pandas as pd\n",
"import numpy as np\n",
"from sklearn.preprocessing import MinMaxScaler\n",
"from sklearn.metrics import confusion_matrix,accuracy_score\n",
"\n",
"from keras.models import Sequential\n",
"from keras.layers import Dense, Dropout, LSTM, Activation\n",
"from keras.callbacks import EarlyStopping\n",
"\n"
},
{
```

"cell_type": "markdown",
"metadata": {
"_uuid": "d730655b568bdaab622ebfb776020c3596f1250f"
},
"source": [
"### Loading Dataset"
]
},
{
"cell_type": "code",
"execution_count": 2,
"metadata": {
"_uuid": "f4550b5117f7c1d9a75281d3b2c013ee08e8eb83"
},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"Shape of Train dataset: (20631, 26)\n"
]
},
{
"data": {
"text/html": [
" \n",
"<style scoped>\n",
"   .dataframe tbody tr th:only-of-type {\n",
"   vertical-align: middle;\n",
"   }\n",
"\n",
"   .dataframe tbody tr th {\n",
"   vertical-align: top;\n",
"   }\n",
"\n",

```
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>id</th>\n",
"      <th>cycle</th>\n",
"      <th>setting1</th>\n",
"      <th>setting2</th>\n",
"      <th>setting3</th>\n",
"      <th>s1</th>\n",
"      <th>s2</th>\n",
"      <th>s3</th>\n",
"      <th>s4</th>\n",
"      <th>s5</th>\n",
"      <th>...</th>\n",
"      <th>s12</th>\n",
"      <th>s13</th>\n",
"      <th>s14</th>\n",
"      <th>s15</th>\n",
"      <th>s16</th>\n",
"      <th>s17</th>\n",
"      <th>s18</th>\n",
"      <th>s19</th>\n",
"      <th>s20</th>\n",
"      <th>s21</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"    </tr>\n",
"  </tbody>\n",
```

```
"</table>\n",
"<p>5 rows × 26 columns</p>\n",
"</div>"
],
"text/plain": [
"   id  cycle  setting1  setting2  setting3      s1      s2      s3      s4  \\\n",
"0   1      1   -0.0007   -0.0004     100.0  518.67  641.82  1589.70  1400.60   \n",
"1   1      2    0.0019   -0.0003     100.0  518.67  642.15  1591.82  1403.14   \n",
"2   1      3   -0.0043    0.0003     100.0  518.67  642.35  1587.99  1404.20   \n",
"3   1      4    0.0007    0.0000     100.0  518.67  642.35  1582.79  1401.87   \n",
"4   1      5   -0.0019   -0.0002     100.0  518.67  642.37  1582.85  1406.22   \n",
"\n",
"      s5   ...       s12     s13     s14     s15   s16  s17   s18    s19  \\\n",
"0  14.62   ...    521.66  2388.02  8138.62  8.4195  0.03  392  2388  100.0   \n",
"1  14.62   ...    522.28  2388.07  8131.49  8.4318  0.03  392  2388  100.0   \n",
"2  14.62   ...    522.42  2388.03  8133.23  8.4178  0.03  390  2388  100.0   \n",
"3  14.62   ...    522.86  2388.08  8133.83  8.3682  0.03  392  2388  100.0   \n",
"4  14.62   ...    522.19  2388.04  8133.80  8.4294  0.03  393  2388  100.0   \n",
"\n",
"     s20      s21  \n",
"0  39.06  23.4190  \n",
"1  39.00  23.4236  \n",
"2  38.95  23.3442  \n",
"3  38.88  23.3739  \n",
"4  38.90  23.4044  \n",
"\n",
"[5 rows x 26 columns]"
]
},
"execution_count": 2,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
```

"data_train=pd.read_csv('PM_train.txt',sep='
',header=None).drop([26,27],axis=1)\n",
"col_names=
['id','cycle','setting1','setting2','setting3','s1','s2','s3','s4','s5','s6','s7','s8','s9','s10','s
11','s12','s13','s14','s15','s16','s17','s18','s19','s20','s21']\n",
"data_train.columns=col_names\n",
"print('Shape of Train dataset: ',data_train.shape)\n",
"data_train.head()"
]
},
{
"cell_type": "code",
"execution_count": 3,
"metadata": {
"_uuid": "aff2e5ec59ea26238a5b87f74b02f2163552541f"
},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"Shape of Test dataset: (13096, 26)\n"
]
},
{
"source": [
"#### Loading Truth table"
]
},
{
"cell_type": "code",
"execution_count": 4,
"metadata": {
"_uuid": "ecb335d1e61debec54d5facfa4d6ae4124a035b0"
},

```
  "source": [
   "## LSTM Network"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 16,
  "metadata": {
   "_uuid": "b125ec1d30bbf3fda850a3b0229ed07e12bc2e32",
   "scrolled": true
  },
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Model: \"sequential\"\n",
     "_________________________________________________________________\n",
     "Layer (type)                 Output Shape              Param #   \n",
     "=================================================================\
n",
     "lstm (LSTM)                  (None, 50, 100)           50000     \n",
     "_________________________________________________________________\n",
     "dropout (Dropout)            (None, 50, 100)           0         \n",
     "_________________________________________________________________\n",
     "lstm_1 (LSTM)                (None, 50)                30200     \n",
     "_________________________________________________________________\n",
     "dropout_1 (Dropout)          (None, 50)                0         \n",
     "_________________________________________________________________\n",
     "dense (Dense)                (None, 1)                 51        \n",
     "=================================================================\
n",
```

```
   "Total params: 80,251\n",
   "Trainable params: 80,251\n",
   "Non-trainable params: 0\n",
   "_________________________________________________________________\n"
  ]
 }
],
"source": [
 "nb_features =X_train.shape[2]\n",
 "timestamp=seq_length\n",
 "\n",
 "model1 = Sequential()\n",
 "\n",
 "model1.add(LSTM(\n",
 "         input_shape=(timestamp, nb_features),\n",
 "         units=100,\n",
 "         return_sequences=True))\n",
 "model1.add(Dropout(0.2))\n",
 "\n",
 "model1.add(LSTM(\n",
 "          units=50,\n",
 "          return_sequences=False))\n",
 "model1.add(Dropout(0.2))\n",
 "\n",
 "model1.add(Dense(units=1, activation='sigmoid'))\n",
 "model1.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])\n",
 "\n",
 "model1.summary()"
]
},
{
 "cell_type": "code",
 "execution_count": 17,
 "metadata": {
  "_uuid": "b8f2d754b02a3917b4ae94696d887b30b60bc3bb"
```

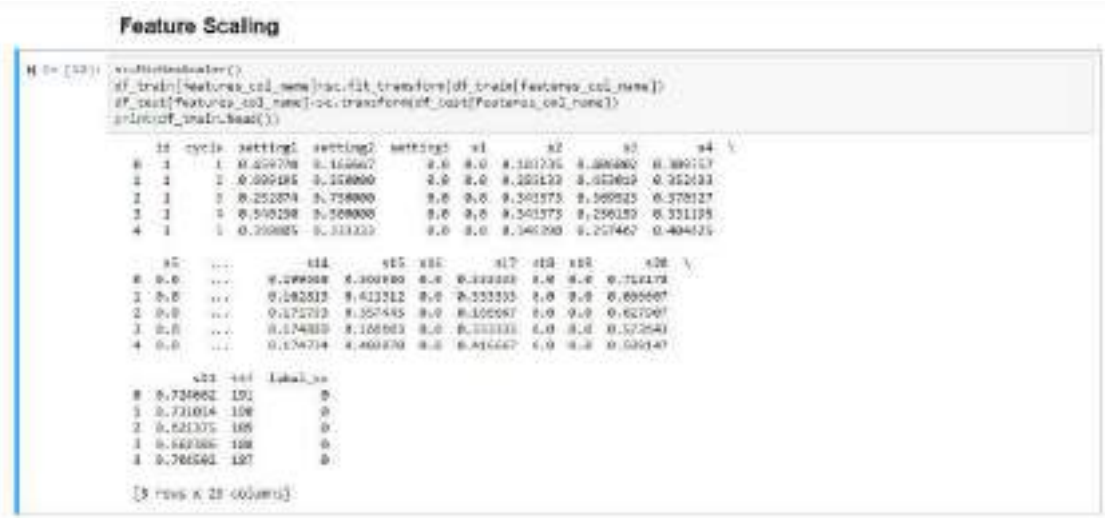
```
},

 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "Probability that machine will fail within 30 days:  0.99172217\n",
   "Maintenance ALERT\n"
  ]
 }
],
"source": [
 "machine_id=100\n",
 "\n",
 "\n",
 "print('Probability that machine will fail within 30 days: ',prob_failure(machine_id))\n",
 "no = prob_failure(machine_id)\n",
 "if no>0.5:\n",
 "    print(\"Maintenance ALERT\")\n",
 "else:\n",
 "    print(\"Machine is alright\")"
]
},
{
 "cell_type": "code",
 "execution_count": 6,
 "metadata": {
  "_uuid": "c4435446bc9bdf3655f8d42df033a3c05bdb5f1f",
  "collapsed": true
 },
 "outputs": [
  {
   "ename": "NameError",
   "evalue": "name 'prob_failure' is not defined",
   "output_type": "error",
```

```
    "traceback": [
   ],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.8.8"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 1
}
```

5.SCREEN SHOTS

5. SCREEN SHOTS

5.1 FEATURE SCALING



5.1 Screenshot: Feature Scaling

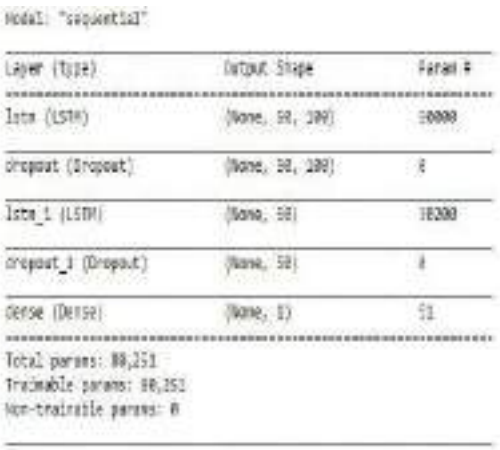
5.2 FUNCTION TO RESHAPE DATASET AS REQUIRED BY LSTM

5.2 Screenshot: Function to reshape dataset as required by LSTM

5.3 BUILDING LSTM NETWORK

5.3 Screenshot: Building LSTM network

5.4 SUMMARY OF LSTM NETWORK

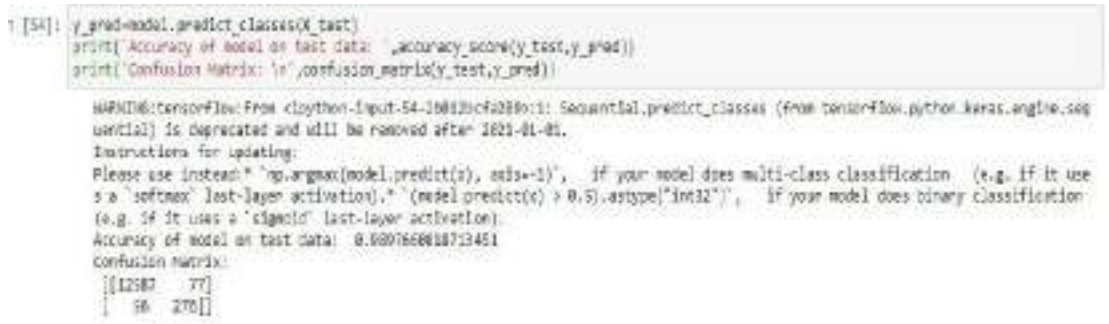


5.4 Screenshot: Summary of LSTM network

5.5 PRE- BUILT MODEL TO FIT TRAINING DATA

5.5 Screenshot: Pre-built model to fit training data

5.6 EVALUATING THE MODEL



5.6 Screenshot: Evaluating the model

5.7 PROBABILITY OF MACHINE FAILURE

5.7 Screenshot: Probability of machine failure

6.TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows, data fields, predefined processes.

6.3 PROBABILITY OF MACHINE FAILURE:

Take machine_id as input. The data of the input machine_id is given to the trained model. The probability that the machine fails within 30 days is calculated by the model. If the probability is greater than 0.5 then “ALERT" is displayed.

Fig. 6.3: Probability of Machine Failure

7.CONCLUSION

7. CONCLUSION

7.1 PROJECT CONCLUSION

We have created a python project which takes input as machine id for which we would like to calculate the probability of failure within a given time period. Time period before which alerts are to be given is decided before training the model. The LSTM neural network is trained using the historic data of NASA Turbofan Jet Engine. The test data of input (machine id) is passed through several layers of LSTM Neural Network. The trained model successfully calculates the probability of failure within the given time period.

7.2 EXPLAIN/DEFINE TERMS

Neural Networks- A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature.

Predictive Maintenance- In Predictive Maintenance, data is collected over time to monitor the state of equipment. The goal is to find patterns that can help predict and ultimately prevent failures. Predictive maintenance will detect the anomalies and failure patterns and provide early warnings.

LSTM- Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems.

Confusion Matrix- A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

8. BIBILOGRAPHY

8. BIBILOGRAPHY

8.1 REFERENCES

[1] https://wiki.pathmind.com/neural-network

[2] https://en.wikipedia.org/wiki/Predictive_maintenance

[3] https://www.ibm.com/cloud/learn/neural-networks

[4] https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

[5] https://towardsdatascience.com/how-to-implement-machine-learning-for-predictive-maintenance-4633cdbe4860

[6] https://www.infoq.com/articles/machine-learning-techniques-predictive-maintenance/

[7] https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-

[8] experts/#:~:text=Long%20Short%2DTerm%20Memory%20(LSTM,complex%20area%20of%20deep%20learning.

[9] https://www.kaggle.com/behrad3d/nasa-cmaps

[10]https://gallery.azure.ai/Experiment/Predictive-Maintenance-Step-1-of-3-data-preparation-and-feature-engineering-

[11]https://tryolabs.com/blog/2020/09/03/predictive-maintenance-using-machine-learning/