

CONTENT

1. INTRODUCTION

- 1.1 Project Overview
- 1.2 Purpose

2. LITERATURE SURVEY

- 2.1 Problem Statement Definition
- 2.2 References

3. IDEATION & PROPOSED SOLUTION

- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

- 4.1 Functional requirement
- 4.2 Non-Functional requirements

5. PROJECT DESIGN

- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule

7. CODING

- 7.1 Final Code

8. SCREENSHOTS

9. ADVANTAGES & DISADVANTAGES

10. CONCLUSION

11. FUTURE SCOPE

12. APPENDIX

GitHub& Project Demo Link

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Engine failure is highly risky and needs a lot of time for repair. Unexpected failure leads to loss of money and time. Predicting the failure prior will save time, effort, money and sometimes even lives. The failure can be detected by installing the sensors and keeping a track of the values. The failure detection and predictive maintenance can be for any device, out of which we will be dealing with the engine failure for a threshold number of days.

The project aims to predict the failure of an engine by using Machine Learning to save loss of time & money thus improving productivity.

1.2 PURPOSE

- To understand the problem to classify if it is a regression or a classification kind of problem.
- To pre-process/clean the data using different data pre-processing techniques.
- To apply different algorithms according to the dataset and will be able to know how to find the accuracy of a model.

You will be able to build web applications using the Flask framework.

2. LITERATURE SURVEY

2.1 PROBLEM STATEMENT DEFINITION:

Problem-Statement: "How might we help the Flight Safety Officer, who wants to improve the engine reliability and flight safety by using a predictive analysis to replace vulnerable components?"

Although the flight safety officer thoroughly inspects the engine before takeoff, he or she cannot guarantee that no failure will occur during flight. To overcome this problem, predictive analytics comes in and predicts the accuracy or gives us prior information on the flight engine. To avoid accidents during flight, this is a good solution to use.

This section includes a literature survey of research on the assessment of student enrollment opportunities in universities.

2.2 REFERENCES

Michael T. Tong Glenn Research Center, Cleveland, Ohio : Machine Learning-Based Predictive Analytics for Aircraft Engine Conceptual Design

<https://ntrs.nasa.gov/api/citations/20205007448/downloads/TM-20205007448.pdf>

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

This map is created with view of the project in user's perspective, to find pain & gain points and to summarize it with a list of problem statements.



3.2 IDEATION & BRAINSTORMING

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes

Nithyalakshmi

- New components, equipment designs and materials can be used.
- Ignition leads can become loose or worn and need periodic replacing.
- Turbochargers maintenance: As it can leak and fittings can become loose.
- Rocker box covers need to be checked as it may become loose and cause oil loss and damage the engine compartment.
- Magneto, which the spark plug cylinders, can become loose and fall off the engine.
- Reducing the drag who makes more aerodynamically efficient which helps in less fuel consumption.
- Oil and fuel fittings and lines should be checked, which can become loose, brittle and cracked. Also cylinder bases, which can leak oil.
- Initially, efficient engine can be used.
- Alternators have drive couplings that can fail and send pieces into the engine causing failure.

Yugalakshmi

- Collect extensive data on working of various engine components.
- Identify replaceable components.
- Identifying the dependencies between components.
- Come up with economic results to promote efficient working of engines.
- To identify and replace a component before it fails.
- Identifying critical components.
- Identifying regular maintenance time period for each component.
- Increasing the life of engine by sending prior notifications on components.
- Scheduling tests on working of components.

Preethi

- Carb heat to clear carburettor icing.
- Examining the level of air intake in preflight inspection.
- Regular testing for microbes and water in stored fuel.
- Analyse thoroughly to prevent fuel starvation.
- Systematic analyses of the symptoms that indicates engine malfunction.
- Battling needs periodic replacements.
- Air filters maintenance: filters that are too old can break down sending shards of filter into the engine choking it.
- Fuel vents can become clogged without regular cleaning.
- Throttle, mixture, prop and carb heat cables can cause engine failure if a cable end fitting or clamp comes loose or undoes.

Reethika

- EHM - Regular updates in engine monitoring systems.
- Test the fails through simulation initially/failures which are measured, analysed, stored where later teach the machine where it can detect those failures priority.
- Maintenance at required intervals.
- Officer can check the engine and that data should be stored before take-off.
- Can add ANN to aircraft systems.
- Automatic temperature maintenance.
- RUL + Regression.

3
Group ideas
Take turns sharing your ideas while choosing similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.
20 minutes

Maintenance

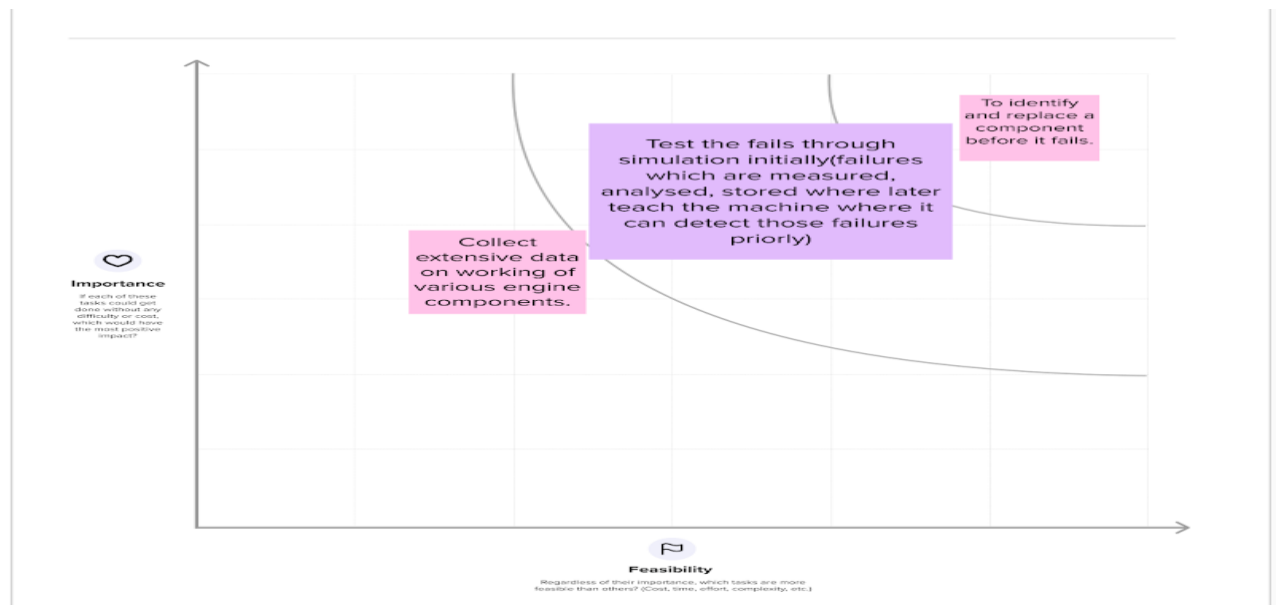
- Air filters maintenance: filters that are too old can break down sending shards of filter into the engine choking it.
- Identifying regular maintenance time period for each component.
- Turbochargers maintenance: As it can leak and fittings can become loose.
- Maintenance at required intervals.
- Scheduling tests on working of components.

Examining and Collecting Data

- EHM - Regular updates in engine monitoring systems.
- Collect extensive data on working of various engine components.
- Examining the level of air intake in preflight inspection.
- Officer can check the engine and that data should be stored before take-off.
- Test the fails through simulation initially/failures which are measured, analysed, stored where later teach the machine where it can detect those failures priority.

Targeting Specific Components

- Ignition leads can become loose or worn and need periodic replacing.
- Carb heat to clear carburettor icing.
- Rocker box covers must be checked as it may become loose and cause oil loss and damage the engine compartment.
- Battling needs periodic replacements.
- Regular testing for microbes and water in stored fuel.
- Magneto, which the spark plug cylinders, can become loose and fall off the engine.
- Fuel vents can become clogged without regular cleaning.
- Oil and fuel fittings and lines should be checked, which can become loose, brittle and cracked. Also cylinder bases, which can leak oil.
- Throttle, mixture, prop and carb heat cables can cause engine failure if a cable end fitting or clamp comes loose or undoes.
- Identifying the dependencies between components.



3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	How might we help the Flight Safety Officer who wants to improve the engine reliability and flight safety by using a predictive analysis to replace vulnerable components.
2.	Idea / Solution description	Collect extensive data on working of engines, develop a Machine Learning Program and train it with failure data generated through a simulation model. The program would then predict on the working of the engine.
3.	Novelty / Uniqueness	Identifying interdependencies between component which would result with the most economical solution.
4.	Social Impact / Customer Satisfaction (<i>FLIGHT SAFETY OFFICER</i>)	The Flight Safety Officer is now able to <i>avoid</i> Engine failure instead of solving the failure after it occurs. Accurate results as a machine can analyse extensive amount of data. The customer is able to produce Engine Assurance quickly.
5.	Business Model (Revenue Model)	The materials, cost of resources and our initial investments are negligible.
6.	Scalability of the Solution	The solution is highly scalable, it can accommodate a large set of data for any type of engine.

3.4 PROBLEM SOLUTION FIT:

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) CS The customer is Flight Safety Officer	6. CUSTOMER CONSTRAINTS CC Scarcity of workforce. Time constraints.	5. AVAILABLE SOLUTIONS AS Manual inspection of the aircraft engines. Implementing the safety procedures plan. But these solutions are timetaking and involve too many permutations and combinations to be considered before tagging it safe to fly.	Explore AS, differentiate
Focus on J&P, tap into BE, understand RC	2. JOBS-TO-BE-DONE / PROBLEMS J&P Emergency response planning. Plan and develop safety procedures. Assist in flight data monitoring. Flight safety officer must have high computer skills. Preventive inspections comes under data monitoring.	9. PROBLEM ROOT CAUSE RC Lack of certain skills to do the job, being lethargic, must have presence of mind to work in emergencies. The backstory behind the need to do this job : Customer has to do jobs to ensure the flight safety.	7. BEHAVIOUR BE Analyses the Flight engine. Reports to the Superior. Always conscious of mishappenings. Promotes the company based on successful journey statistics.	Focus on J&P, tap into BE, understand RC
Identify strong TR & EM	3. TRIGGERS TR Seeing other aviation companies using our product which also results in less engine failures/flight accidents. More the data analysis done by the machine, less work for the officer.	10. YOUR SOLUTION SL Collect extensive data on working of engines, develop a Machine Learning Program and train it with failure data generated through a simulation model. The program would then predict on the working of the engine. The Flight Safety Officer is now able to avoid engine failure instead of solving the failure after it occurs. The customer is able to produce engine assurance quickly.	8. CHANNELS OF BEHAVIOUR CH 8.1 ONLINE Reports to the superior via mails. 8.2 OFFLINE Reports to the superior in person. Monitors subordinates.	Identify strong TR & EM
	4. EMOTIONS: BEFORE / AFTER EM When facing a problem the customer is under pressure and feels perturbed. After solving it they feel relieved and are confident that there wouldn't be any complaints from the manual inspector. They are happy as they expect zero failure now.			

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS:

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The flight safety officer is provided with an undisturbed ambience, a system with a condition for the officer to perform the tasks safely, effectively, and efficiently.
NFR-2	Security	Assuring all data inside the system or its part will be protected against malware attacks or unauthorized access.
NFR-3	Reliability	A good working system with adequate ML datasets that predict engine failure.
NFR-4	Performance	It defines how fast a software system or a particular piece of it responds to the flight safety officer. This metric explains how long a user must wait before the target operation happens given the overall number of users at the moment. As backup will be provided during the process, performance is adequate.
NFR-5	Availability	Highly available.
NFR-6	Scalability	The solution is highly scalable, it can accommodate a large set of data for any type of engine

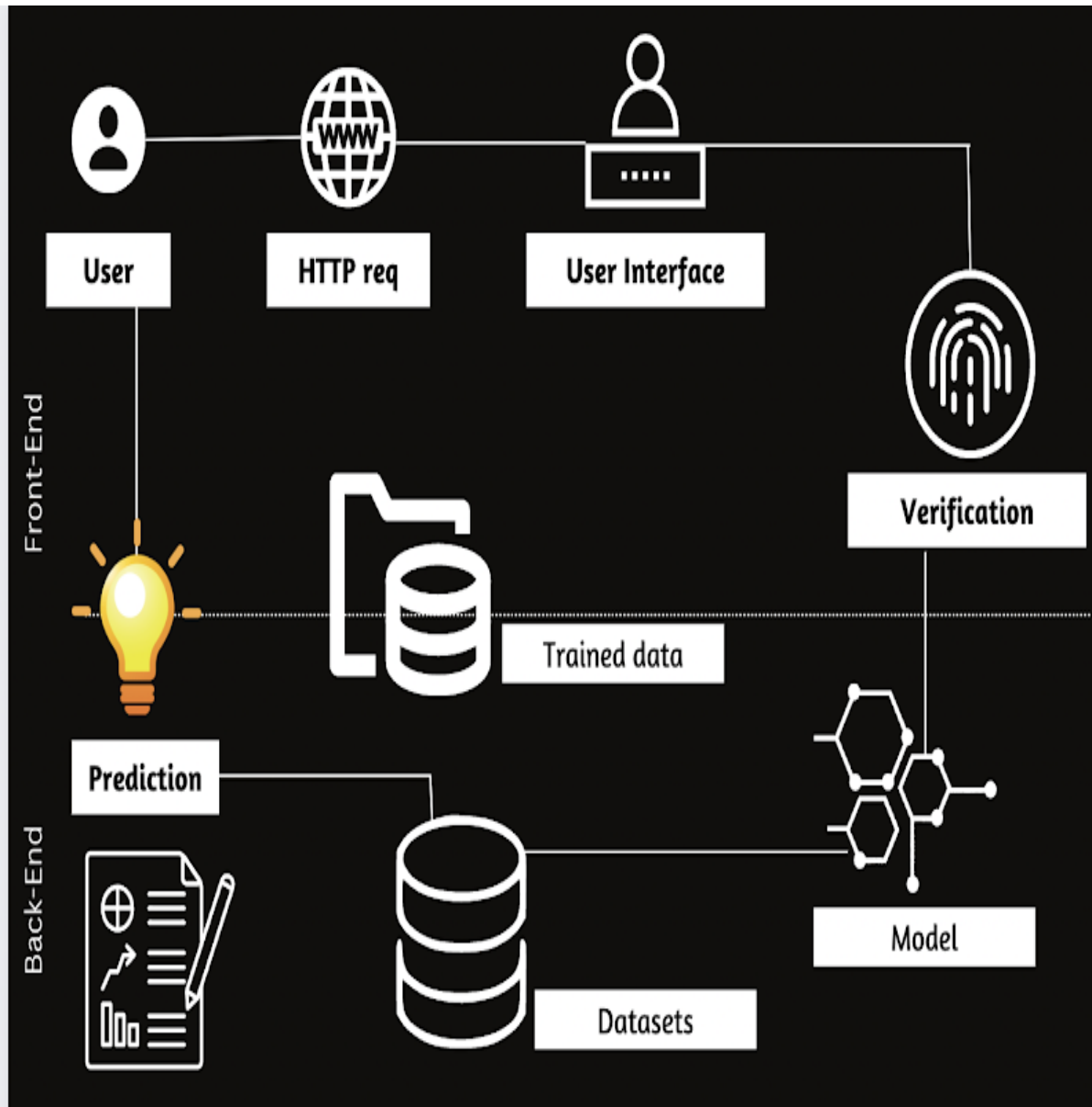
4.2 NON-FUNCTIONAL REQUIREMENTS:

FR No.	Functional Requirement (Epic) <i>[User – Flight Safety Officer]</i>	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through website
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Verification	Fingerprint Verification
FR-4	Inputs/Data	Data generated through a simulation model and engine data.
FR-5	Data processing	Displays the components lifespan, detects errors.
FR-6	Prediction	Notifies when there's a flaw in components and displays the engine failure rate.

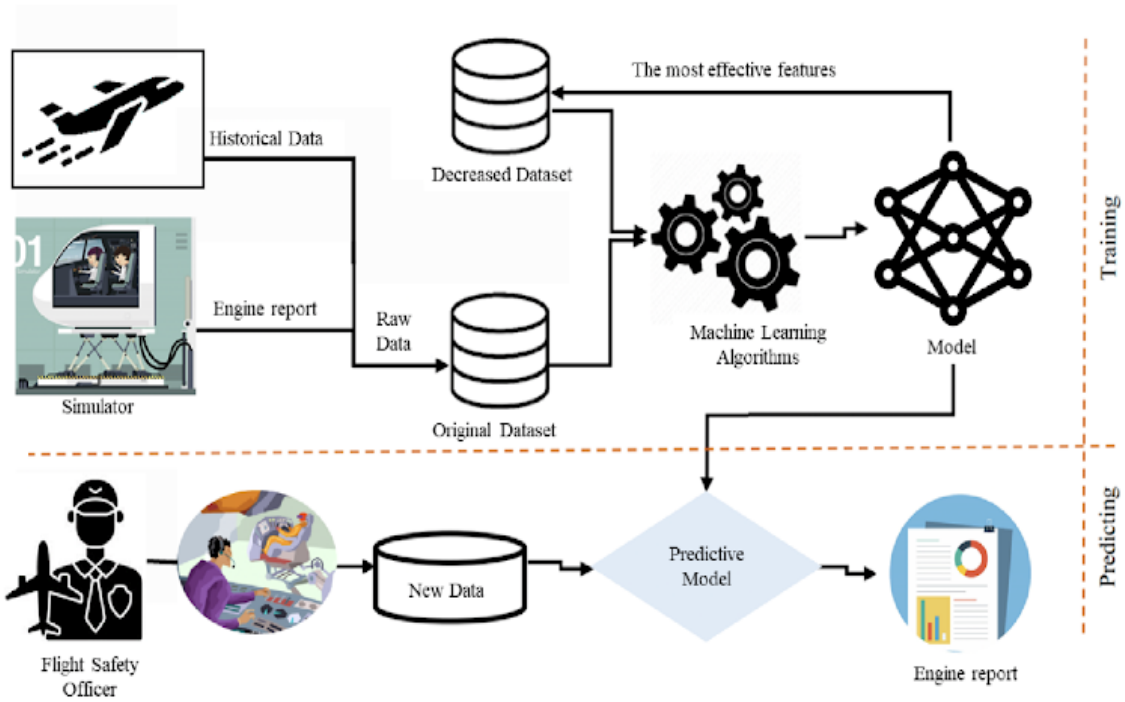
5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 SOLUTION ARCHITECTURE:



5.2 USER STORIES:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web User)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web Verification)	Fingerprint biometric sensor		As a user, I can verify through my fingerprint for security purpose	For authentication	Medium	Sprint 1
Customer Care Executive	Customer service		If any issues arises	Customer Care Executive will solve the issues/errors		

6.PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022		19 Nov 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	19 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	19 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 1	Modules	USN-1	The module which are used to be imported	2	High	NithyaLakshmi Yugalakshmi
Sprint - 1	Data Ingestion	USN-2	Instead of a CSV file, text file is added	1	High	Preethi Yugalakshmi
Sprint - 2	Data Pre-processing	USN-3	A data mining technique	2	Medium	Reethika NithyaLakshmi
Sprint - 3	Modelling	USN-4	Data Modeling in software engineering is the process of simplifying the diagram or data model of a software system by applying certain formal techniques.	2	Medium	Yugalakshmi Preethi
Sprint - 4	LSTM Network	USN-5	LSTMs are predominately used to learn, process, and classify sequential data	1	High	Preethi Reethika
Sprint - 4	Model Training	USN-6	Training the model		High	Nithyalakshmi Yugalakshmi Preethi Reethika
Sprint - 4	Model Testing	USN-7	Test and validation of model	2	High	Preethi Reethika

7. CODING

7.1 FINAL CODE:

```
1 import keras
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 np.random.seed(1234)
6 PYTHONHASHSEED = 0
7 from sklearn import preprocessing
8 from sklearn.metrics import confusion_matrix, recall_score, precision_score
9 from keras.models import Sequential
10 from keras.layers import Dense, Dropout, LSTM, Activation
11 %matplotlib inline
12
13 # read training data
14 train_df = pd.read_csv('PM_train.txt', sep=" ", header=None)
15 train_df.drop(train_df.columns[[26, 27]], axis=1, inplace=True)
16 train_df.columns = ['id', 'cycle', 'setting1', 'setting2', 'setting3', 's1', 's2', 's3',
17                    's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's14',
18                    's15', 's16', 's17', 's18', 's19', 's20', 's21']
19
20 # read test data
21 test_df = pd.read_csv('PM_test.txt', sep=" ", header=None)
22 test_df.drop(test_df.columns[[26, 27]], axis=1, inplace=True)
23 test_df.columns = ['id', 'cycle', 'setting1', 'setting2', 'setting3', 's1', 's2', 's3',
24                   's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's14',
25                   's15', 's16', 's17', 's18', 's19', 's20', 's21']
26
27
28 # read ground truth data
29 truth_df = pd.read_csv('PM_truth.txt', sep=" ", header=None)
30 truth_df.drop(truth_df.columns[[1]], axis=1, inplace=True)
31
32 train_df = train_df.sort_values(['id', 'cycle'])
33 train_df.head()
34
35 rul = pd.DataFrame(train_df.groupby('id')['cycle'].max()).reset_index()
36 rul.columns = ['id', 'max']
37 train_df = train_df.merge(rul, on=['id'], how='left')
38 train_df['RUL'] = train_df['max'] - train_df['cycle']
39 train_df.drop('max', axis=1, inplace=True)
40 train_df.head()
```

```

79 test_df.head()
80
81 test_df['label1'] = np.where(test_df['RUL'] <= w1, 1, 0 )
82 test_df['label2'] = test_df['label1']
83 test_df.loc[test_df['RUL'] <= w0, 'label2'] = 2
84 test_df.head()[test_df['label1'] = np.where(test_df['RUL'] <= w1, 1, 0 )
85 test_df['label2'] = test_df['label1']
86 test_df.loc[test_df['RUL'] <= w0, 'label2'] = 2
87 test_df.head()
88
89 sequence_length = 50
90
91 engine_id3 = test_df[test_df['id'] == 3]
92 engine_id3_50cycleWindow = engine_id3[engine_id3['RUL'] <= engine_id3['RUL'].min() + 50]
93 cols1 = ['s1', 's2', 's3', 's4', 's5', 's6', 's7', 's8', 's9', 's10']
94 engine_id3_50cycleWindow1 = engine_id3_50cycleWindow[cols1]
95 cols2 = ['s11', 's12', 's13', 's14', 's15', 's16', 's17', 's18', 's19', 's20', 's21']
96 engine_id3_50cycleWindow2 = engine_id3_50cycleWindow[cols2]
97
98 ax1 = engine_id3_50cycleWindow1.plot(subplots=True, sharex=True, figsize=(20,20))
99 ax2 = engine_id3_50cycleWindow2.plot(subplots=True, sharex=True, figsize=(20,20))
100
101 def gen_sequence(id_df, seq_length, seq_cols):
102     data_array = id_df[seq_cols].values
103     num_elements = data_array.shape[0]
104     for start, stop in zip(range(0, num_elements-seq_length), range(seq_length, num_elements)):
105         yield data_array[start:stop, :]
106
107
108
109 sensor_cols = ['s' + str(i) for i in range(1,22)]
110 sequence_cols = ['setting1', 'setting2', 'setting3', 'cycle_norm']
111 sequence_cols.extend(sensor_cols)
112 seq_gen = (list(gen_sequence(train_df[train_df['id']==id], sequence_length, sequence_cols))
113            for id in train_df['id'].unique())
114
115 seq_array = np.concatenate(list(seq_gen)).astype(np.float32)
116 seq_array.shape

```

```

155 y_pred = model.predict_classes(seq_array, verbose=1, batch_size=200)
156 y_true = label_array
157 print('Confusion matrix\n- x-axis is true labels.\n- y-axis is predicted labels')
158 cm = confusion_matrix(y_true, y_pred)
159 print(cm)
160
161 precision = precision_score(y_true, y_pred)
162 recall = recall_score(y_true, y_pred)
163 print('precision = ', precision, '\n', 'recall = ', recall)
164 seq_array_test_last = [test_df[test_df['id']==id][sequence_cols].values[-sequence_length:]
165                        for id in test_df['id'].unique() if len(test_df[test_df['id']==id]) >= sequence_length]
166
167 seq_array_test_last = np.asarray(seq_array_test_last).astype(np.float32)
168 seq_array_test_last.shape
169
170 y_mask = [len(test_df[test_df['id']==id]) >= sequence_length for id in test_df['id'].unique()]
171
172 label_array_test_last = test_df.groupby('id')['label1'].nth(-1)[y_mask].values
173 label_array_test_last = label_array_test_last.reshape(label_array_test_last.shape[0],1).astype(np.float32)
174 label_array_test_last.shape
175
176 print(seq_array_test_last.shape)
177 print(label_array_test_last.shape)
178 scores_test = model.evaluate(seq_array_test_last, label_array_test_last, verbose=2)
179 print('Accuracy: {}'.format(scores_test[1]))
180
181 y_pred_test = model.predict_classes(seq_array_test_last)
182 y_true_test = label_array_test_last
183 print('Confusion matrix\n- x-axis is true labels.\n- y-axis is predicted labels')
184 cm = confusion_matrix(y_true_test, y_pred_test)
185 print(cm)
186
187 precision_test = precision_score(y_true_test, y_pred_test)
188 recall_test = recall_score(y_true_test, y_pred_test)
189 f1_test = 2 * (precision_test * recall_test) / (precision_test + recall_test)
190 print('Precision: ', precision_test, '\n', 'Recall: ', recall_test, '\n', 'F1-score:', f1_test )

```



```

180
181 y_pred_test = model.predict_classes(seq_array_test_last)
182 y_true_test = label_array_test_last
183 print('Confusion matrix\n- x-axis is true labels.\n- y-axis is predicted labels')
184 cm = confusion_matrix(y_true_test, y_pred_test)
185 print(cm)
186
187 precision_test = precision_score(y_true_test, y_pred_test)
188 recall_test = recall_score(y_true_test, y_pred_test)
189 f1_test = 2 * (precision_test * recall_test) / (precision_test + recall_test)
190 print('Precision: ', precision_test, '\n', 'Recall: ', recall_test, '\n', 'F1-score:', f1_test)
191
192 results_df = pd.DataFrame([scores_test[1], precision_test, recall_test, f1_test],
193                           [0.94, 0.952381, 0.8, 0.869565]),
194                           columns = ['Accuracy', 'Precision', 'Recall', 'F1-score'],
195                           index = ['LSTM',
196                                   'Template Best Model'])
197
198 print(results_df)
199
200
201
202

```

8 SCREENSHOTS:

The screenshot shows a Jupyter Notebook interface with the following content:

```

Precision: 0.96
Recall: 0.96
F1-score: 0.96

```

```

results_df = pd.DataFrame([scores_test[1], precision_test, recall_test, f1_test],
                           [0.94, 0.952381, 0.8, 0.869565]),
                           columns = ['Accuracy', 'Precision', 'Recall', 'F1-score'],
                           index = ['LSTM',
                                   'Template Best Model'])
results_df

```

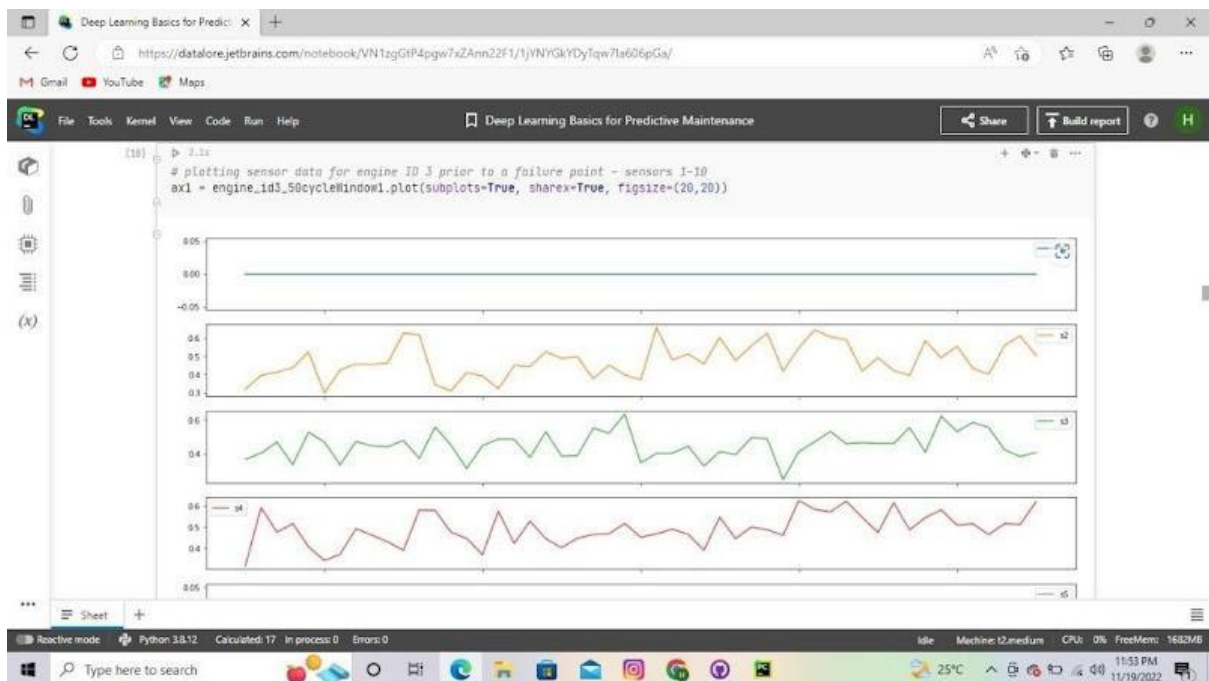
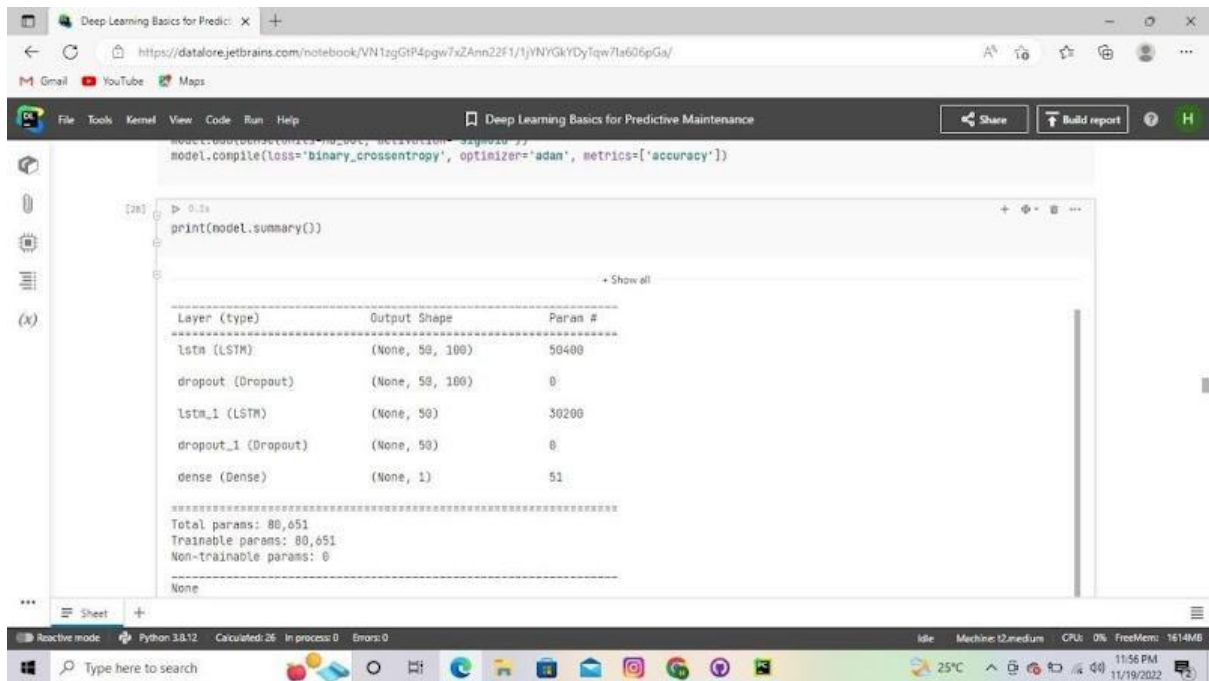
	Accuracy	Precision	Recall	F1-score
LSTM	0.978495	0.960000	0.86	0.960000
Template Best Model	0.940000	0.952381	0.80	0.869565

```

Accuracy Precision Recall F1-score
LSTM 0.978495 0.960000 0.86 0.960000
Template Best Model 0.940000 0.952381 0.80 0.869565

```

The interface also shows the file explorer on the left, the top navigation bar with 'Deep Learning Basics for Predictive Maintenance', and the bottom status bar with system information.



9 ADVANTAGES :

Integration with IOT devices to address specific business problem related to Engine.

- Ability to view Engine Component Health.
- The solution can be used / deployed on PC or an iOS device (iPad).
- Ability to create Maintenance Work Orders in S/4 HANA or Gen2iMRO
- Integration with R to utilize its clustering algorithms; Uses PdMS / PAL (Predictive Analysis Library) in SAP HANA for RUL calculation Instead of R model to utilize the algorithms available in PAL (Predictive Analysis Library) in SAP HANA
- Proactive alerts based on user defined parameters Supports Proactive Maintenance of Engines to ensure that the Aircrafts have maximum Airtime.
- Ability to Predict RUL (Remaining Useful Life) of an Aircraft Engine by measuring key parameters like Temperature, Pressure, Physical Fan speed, By Pass Ratio to predict remaining usable life of an Aircraft Engine.
- Reduction in Revenue loss by predicting failures of Aircraft Engines and optimizing service windows.

10 CONCLUSION:

In aviation, the use of maintenance data is highly critical in the analysis of reliability and maintenance costs. It is because predictive maintenance scheduling can be planned in line with estimates. Main target of predictive maintenance is to predict equipment failures and planning strategies for spare parts of the system components to analyse the reliability and maintainability of a complex repairable system. In this study, a hybrid data preparation model was applied to the landing gear system maintenance dataset using feature selection ReliefF

algorithm to select attributes and a modified K-means algorithm to eliminate noisy and inconsistent data. Proposed hybrid data preparation method was put into practice through LR, SVR, and MLP models. Results indicated that the LR model had better performance than MLP and SVR models in predicting the failure counts. Results indicate that the proposed hybrid data preparation model significantly improves the accurate prediction of failure counts. This study could function as a guide for using hybrid data preparation methods in machine learning algorithms and data mining.

11 FUTURE SCOPE:

Our recent experience with players in this space, however, suggests the opposite. While certain “moonshot” (unproven, capital-intensive) predictive maintenance efforts may be less likely, many airlines and MRO organizations, seeking to avoid building back overhead organizations that were sharply reduced as air travel plummeted, are now even more eager to find cost-efficient ways to improve maintenance performance.

Many airlines have already, in recent years, benefited from the use of AI in engine health monitoring (EHM) and have embarked on program optimization campaigns designed to contain costs and increase aircraft reliability. But relatively few have succeeded in early attempts to apply AI to the much more complex component

and line maintenance environments. We believe the predictive power of AI could contribute to a step change in airline profitability by removing maintenance-caused disruption from the operation.

The airlines already exploring the potential of predictive maintenance have found their attempts plagued by inadequate sponsorship at the executive level,

data fragmentation and usability issues, and myriad downstream implementation challenges.

