Predictive Analysis Of Aircraft Engine

Submitted by

Manu P S

Robin Baby

INTRODUCTION

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.

1.1 Overview

Aircrafts are very important part of modern age. The number of passengers traveling by airplanes has been increasing every yearIt is crucial that Aircraft Engines should undergo proper maintenance. Doing a routine maintenance can be very expensive. Predictive maintenance is an effective alternative to it. This approach ensures cost saving. It is also called as condition-based maintenance, as the degrading state of an item is estimated to schedule a maintenance. Machine/Deep Learning are widely used for predictive maintenance.

1.2 Purpose

The Project Aims to reduce failure of aircraft engines and to solve any unconditional changes happens in the system.Mainly used to predict is there any chance for an engine failure. Depending upon the airplane, an engine failure may or may not lead to catastrophic events that end in families filing suit in aviation litigation cases.Every human value is precious and cannot be lost.even any minute mistake can cause engine failure and we need to observe every parts of the system before a take off.if any mismatch occurs a prediction can be made from sample dataset.this would help to solve engine failure.

LITERATURE SURVEY

2.1 Existing problem

The existing system is the we should have home sample datset collected from differnet Aircrafts and study their data and take necessary actions to avoid engine failure.Collects data from Different types of engine.Huge volume of data.high computation power is needed to process the data.Aircraft engines are of different types like turbo engine,hydraulic engine etc..Each of these engines have different performance power and their failure differ from each other.so easy comparison between these engine data is not easy.

2.2 Proposed solution

The proposed system overcomes all the disadvantages of existing system.here it is based on sensor data.that is different types of sensors are use to track the performance of aircraft engines and easy to predict the failure if any system hanged.easy to handle than manual data that provide.sensor data senses and predict the engine failure prediction. f a failure is detected, sensors can alert administrators and sometimes retroactively disable machines. In many cases, this can happen before the failure actively starts, preventing company downtime and potential employee injury. Consistent manual checks.

3.THEORITICAL ANALYSIS

3.1 Hardware / Software designing

Hardware and software requirements of the project

Hardware Requirements

Processor : Intel CoreTM i5-9300H

Processor speed : 2.4GHz

RAM Size : 4 GB

System Type : X64-based processor

SOFTWARE DESIGNING:

The software required for the development of this project is:

Desktop GUI : Anaconda Navigator

Operating system : Windows 10

Front end : HTML, CSS,JAVASCRIPT

Programming : PYTHON

Cloud Computing Service : IBM Cloud Services

4 EXPERIMENTAL INVESTIGATIONS

IMPORTING AND READING THE DATASET

Importing the Libraries

First step is usually importing the libraries that will be needed in the program.

Pandas: It is a python library mainly used for data manipulation.

NumPy: This python library is used for numerical analysis.

Matplotlib and Seaborn: Both are the data visualization library used for plotting graph which will help us for understanding the data.

csr_matrix() :A dense matrix stored in a NumPy array can be converted into a sparse matrix using the CSR representation by calling the csr_matrix() function.

Train_test_split: used for splitting data arrays into training data and for testing data.

Pickle: to serialize your machine learning algorithms and save the serialized format to a file.

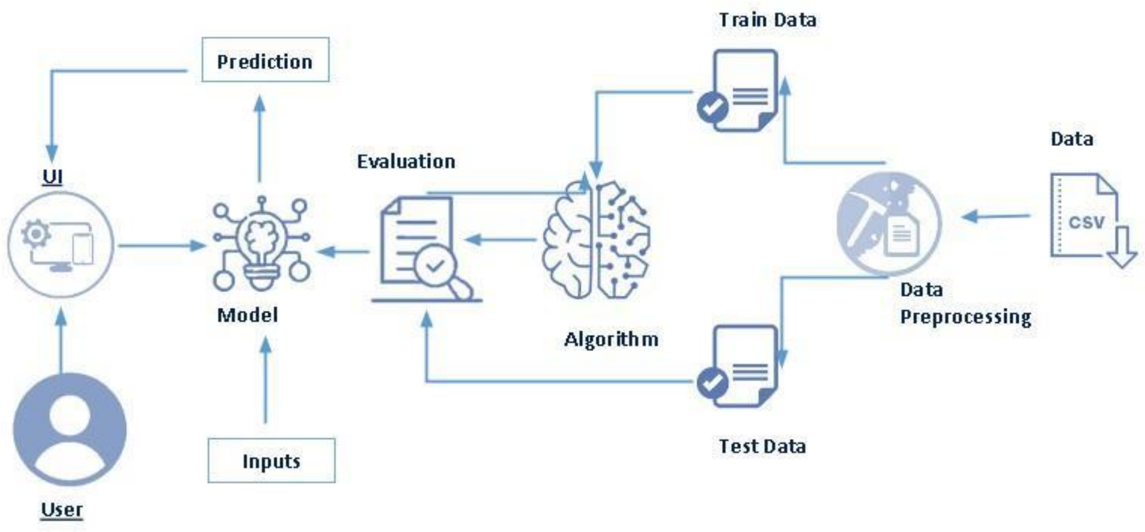
Reading the Dataset

For this project, we make use of three different datasets (Books_Ratings, Books, Users). We will be selecting the important features from these datasets that will help us in recommending the best results.

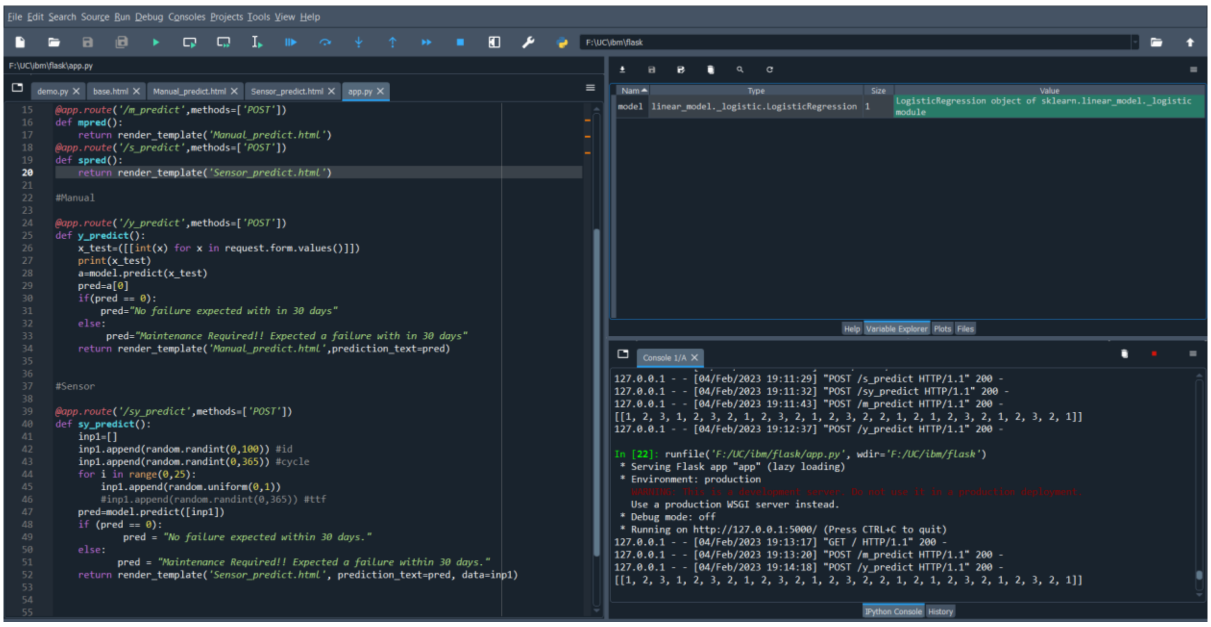
The next step is to read the dataset into a data structure that's compatible with pandas. Let's load a .csv data file into pandas. There is a function for it, called read_csv().We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program).If the dataset in same directory of your program, you can directly read it, without any path. After the next Steps we made following bellow:

- Pre-process the dataset
- Calculate time to failure
- Split the dataset
- Model building
- Application building
- Run the application

5 FLOWCHART



6 RESULT

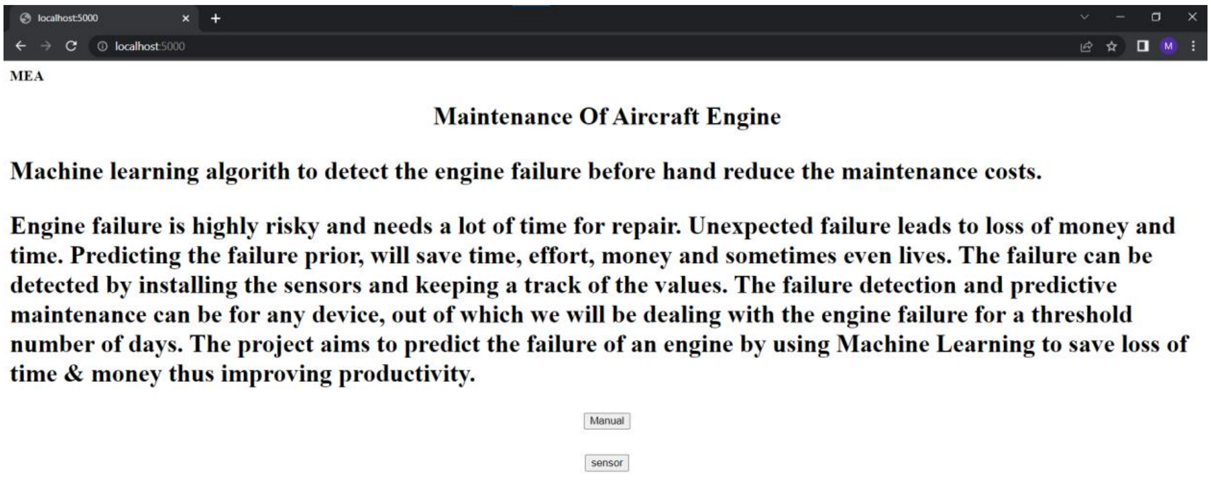

F:\UC\ibm\flask\app.py

```
@app.route('/m_predict',methods=['POST'])
def mpred():
    return render_template('Manual_predict.html')
@app.route('/s_predict',methods=['POST'])
def spred():
    return render_template('Sensor_predict.html')

#Manual

@app.route('/y_predict',methods=['POST'])
def y_predict():
    x_test=([[int(x) for x in request.form.values()]])
    print(x_test)
    a=model.predict(x_test)
    pred=a[0]
    if(pred == 0):
        pred="No failure expected with in 30 days"
    else:
         pred="Maintenance Required!! Expected a failure with in 30 days"
    return render_template('Manual_predict.html',prediction_text=pred)


#Sensor

@app.route('/sy_predict',methods=['POST'])
def sy_predict():
    inp1=[]
    inp1.append(random.randint(0,100)) #id
    inp1.append(random.randint(0,365)) #cycle
    for i in range(0,25):
        inp1.append(random.uniform(0,1))
        #inp1.append(random.randint(0,365)) #ttf
    pred=model.predict([inp1])
    if (pred == 0):
            pred = "No failure expected within 30 days."
    else:
            pred = "Maintenance Required!! Expected a failure within 30 days."
    return render_template('Sensor_predict.html', prediction_text=pred, data=inp1)
```

Nam	Type	Size	Value
model	linear_model._logistic.LogisticRegression	1	LogisticRegression object of sklearn.linear_model._logistic module

```
127.0.0.1 - - [04/Feb/2023 19:11:29] "POST /s_predict HTTP/1.1" 200 -
127.0.0.1 - - [04/Feb/2023 19:11:32] "POST /sy_predict HTTP/1.1" 200 -
127.0.0.1 - - [04/Feb/2023 19:11:43] "POST /m_predict HTTP/1.1" 200 -
[[1, 2, 3, 1, 2, 3, 2, 1, 2, 3, 2, 1, 2, 3, 2, 2, 1, 2, 1, 2, 3, 2, 1, 2, 3, 2, 1]]
127.0.0.1 - - [04/Feb/2023 19:12:37] "POST /y_predict HTTP/1.1" 200 -

In [22]: runfile('F:/UC/ibm/flask/app.py', wdir='F:/UC/ibm/flask')
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [04/Feb/2023 19:13:17] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [04/Feb/2023 19:13:20] "POST /m_predict HTTP/1.1" 200 -
127.0.0.1 - - [04/Feb/2023 19:14:18] "POST /y_predict HTTP/1.1" 200 -
[[1, 2, 3, 1, 2, 3, 2, 1, 2, 3, 2, 1, 2, 3, 2, 2, 1, 2, 1, 2, 3, 2, 1, 2, 3, 2, 1]]
```

Fig 6.1 Flask code on spyder



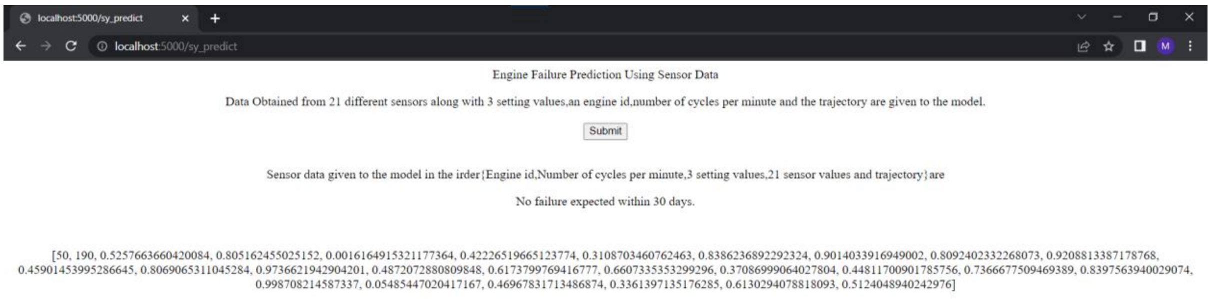
MEA

Maintenance Of Aircraft Engine

Machine learning algorith to detect the engine failure before hand reduce the maintenance costs.

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.

Fig 6.2 Home page



Engine Failure Prediction Using Sensor Data

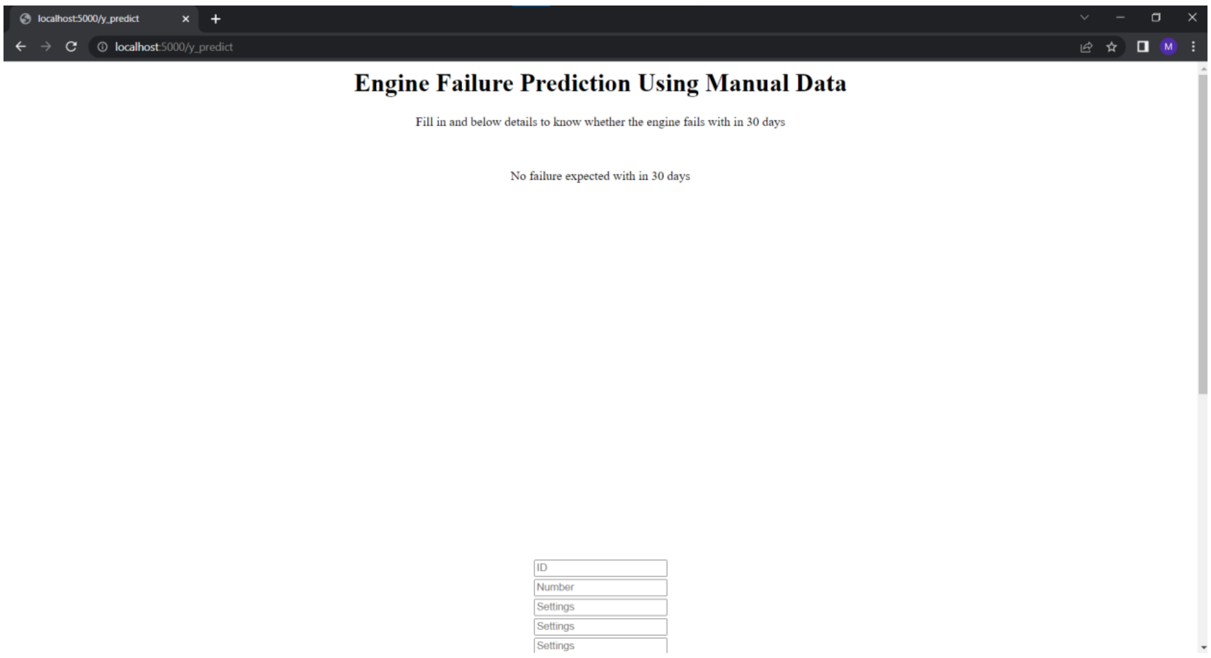
Data Obtained from 21 different sensors along with 3 setting values,an engine id,number of cycles per minute and the trajectory are given to the model.

Sensor data given to the model in the irder{Engine id,Number of cycles per minute,3 setting values,21 sensor values and trajectory}are

No failure expected within 30 days.

```
[50, 190, 0.5257663660420084, 0.805162455025152, 0.0016164915321177364, 0.42226519665123774, 0.3108703460762463, 0.8386236892292324, 0.9014033916949002, 0.8092402332268073, 0.9208813387178768,
0.45901453995286645, 0.8069065311045284, 0.9736621942904201, 0.4872072880809848, 0.6173799769416777, 0.6607335353299296, 0.37086999064027804, 0.44811700901785756, 0.7366677509469389, 0.8397563940029074,
0.998708214587337, 0.05485447020417167, 0.46967831713486874, 0.3361397135176285, 0.6130294078818093, 0.5124048940242976]
```

Fig 6.3 Sensor Page



Engine Failure Prediction Using Manual Data

Fill in and below details to know whether the engine fails with in 30 days

No failure expected with in 30 days

Fig 6.4 Manual Page

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- Able to predict engine failures
- Using sensors track the engine performance
- Need to handle less sensor data compared to manual data

DISADVANTAGES

- Need to check every portion of the engine before take off
- Need to handle huge data
- Manualy need to check the performance of engine
- Predict failure using failurered dataset

8.APPLICATIONS

Applicable in all area where the engine failure can be predicted using sensor data.Used in all vehicles and valuates the performsnce of the engine.different types of sensors are availbale to sense the performance of engines .it is easy for vehicle industry to manage and avoid accidents due to engine failure.

9.CONCLUSION

In aviation, the use of maintenance data is highly critical in the analysis of reliability and maintenance costs. This is because predictive maintenance scheduling can be planned in line with estimates. The main target of predictive maintenance is to predict engine failures and planning strategies for spare parts of the system components to analyze the reliability and maintainability of a complex repairable system.

10 FUTURE SCOPE

Available in all vehicles and easy to manage the vehicle and evaluates their performance without manually.

11 BIBILOGRAPHY

 Hastie, Friedman, and Tibshirani, The Elements of Statistical Learning,2001

 Bishop, Pattern Recognition and Machine Learning, 2006

 Ripley, Pattern Recognition and Neural Networks, 1996

 Duda, Hart, and Stork, Pattern Classification, 2nd Ed., 2002

 Tan, Steinbach, and Kumar, Introduction to Data Mining, Addison-Wesley,2005.

 Scholkopf and Smola, Learning with Kernels, 2002

APPENDIX

A.Source Code

```
from flask import Flask, render_template, request,redirect
import random
import numpy as np
import pickle
import pandas as pd
model = pickle.loads(open("engine_model.pkl",'rb').read())
app=Flask(__name__)
@app.route("/")
def home():
	 return render_template('base.html')
@app.route('/m_predict',methods=['POST'])
def mpred():
	return render_template('Manual_predict.html')
@app.route('/s_predict',methods=['POST'])
def spred():
	return render_template('Sensor_predict.html')
#Manual
```

```
@app.route('/y_predict',methods=['POST'])
def y_predict():
        x_test=([[int(x) for x in request.form.values()]])
        print(x_test)
        a=model.predict(x_test)
        pred=a[0]
        if(pred == 0):
            pred="No failure expected with in 30 days"
        else:
            pred="Maintenance Required!! Expected a failure with in 30 days"
        return render_template('Manual_predict.html',prediction_text=pred)

#Sensor
@app.route('/sy_predict',methods=['POST'])
def sy_predict():
        inp1=[]
        inp1.append(random.randint(0,100)) #id
        inp1.append(random.randint(0,365)) #cycle
        for i in range(0,25):
        inp1.append(random.uniform(0,1))
        #inp1.append(random.randint(0,365)) #ttf
        pred=model.predict([inp1])
        if (pred == 0):
              pred = "No failure expected within 30 days."
        else:
              pred = "Maintenance Required!! Expected a failure within 30 days."
        return render_template('Sensor_predict.html', prediction_text=pred, data=inp1)
if __name__ == '__main__':
        app.run(debug=False)
```