Fault prediction in aircraft engines using Regularized Greedy Forests.

Abstract- Aircraft engines maintenance is very challenging and costly task. The main objective is to ensure a proper operation of the aircraft engines, in all conditions, with a zero probability of engine failure, while taking into account aging of the aircraft. Aircraft maintenance can be improved if an efficient procedure for the prediction of failures is implemented. Several variables such as the core speed, the pressure , the fan speed of the aircraft , etc. are measured and taken in to consideration while predicting the faults , together with environmental variables such as the outside temperature, altitude, aircraft speed, etc. In this paper we demonstrate the application of regularized greedy forest algorithm which directly learns decision forests via fully-corrective regularized greedy search using the underlying forest structure on an aircraft engine data involving failures of aircraft engines. The prediction accuracy obtained with the Regularized Greedy Forests presents a significant improvement over the state-of-the-art ensemble classifiers. Moreover, the performance of three ensemble classifiers with different characteristics – Random Forest, Regularized Greedy Forest, and XGBoost- was compared in terms of their prediction accuracy. We achieved higher accuracy and smaller models using regularized greedy forests on jet engine data set compared to state of the art ensemble methods.

***Index Terms*: Aircraft engine maintenance, Fault prediction, Random Forest, Regularized Greedy Forest, XGBoost, Normalization, and Recursive Feature Elimination.**

CONTENTS

I. INTRODUCTION

Ensuring a proper operation of aircraft engines over their lifetime is an important task.
Aircraft engines are built with a high level of security. They undergo regularly a full maintenance with disassembling, replacement of parts, etc. In addition, between two such maintenances, many parameters are measured on the engines during the flights. These parameters are recorded, and used both at short and long terms for immediate action and alarm generation respectively [1].
In this paper, we are interested in the prediction of engines failure of an aircraft in different time windows.

We present and compare data-driven approaches focusing on ensemble based classifiers used in classification of engine failure in different time windows to the next planned service visit. The model is based on the data collected from sources: Turbofan engine data set from NASA repository.
Regularized greedy forest [6], Random Forests [3] and XGBoost [4] ensemble based algorithms are used for Predicting if an engine will fail in different time windows: E.g., fails in window [1, *w0*] days; fails in the window [*w0*+1, *w1*] days; not fail within *w1* days.
The following of this paper is organized as follows. In Section 2, the data are described and notations are defined. Section 3 presents the methodology: Section 4 describes the experimental results and section 5 describes the conclusion and future work and section 6 describes the references.

II. EXPERIMENT DATA SET

The dataset in this experiment was used for the prognostics challenge competition at the International Conference on Prognostics and Health Management (PHM08) and can be obtained from NASA Ames Prognostics Data Repository [7][12].

It consists of multiple multivariate time series with "cycle" as the unit, together with 21 sensor readings for each cycle. Each time series can be considered 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 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 it is regarded that the engine is not safe to be operated any more. In other words, the last cycle in each time series can be considered as the failure point of the corresponding engine.

III. METHODOLOGY

The Turbo engine data set which is publically available from the NASA repository is split in to three files for the further analysis

A. Training data set – compromises of the aircraft engine run-to-failure data.

B. Testing data set - It is the aircraft engine operating data without failure events recorded.

C. Ground truth data: made up of the information of true remaining cycles for each engine in the testing data.

3.1 Data preprocessing and Feature engineering:-

a) Firstly the data set is checked for any missing values and treated using the last observations carried forward method based on the sampling frequency of sensors in the data set

b) Exploratory data analysis is performed to capture the distribution of the various sensor parameters.

c) Data set is normalized to have zero mean and unit variance.

d) Data quality check is performed to remove zero or unit variance sensors for further analysis.

e) Most of the sensor parameter are highly correlated and such redundant parameters are removed using correlation analysis.

f) Feature engineering and transformation is done on the raw sensor data set and various advanced statistical functions, as well as advanced mathematical functions were are applied to get the aggregated and transformed sensors data.

g) The time window of prediction for parameters *w0 and w1* are predefined. The user's needs to decide how far ahead of time the alert of failure should trigger before the actual failure event.

Finally we have derived the variable to be predicted:-whether engine is going to fail within the window [1, *w0*] days or to fail within the window [*w0*+1, *w1*] days, or it will not fail within *w1* days and attached to the training data set. Value of *w0 =15 and w1=30 days in our current analysis*

3.2 Feature Selection

a) Most frequently used feature selection technique Recursive feature elimination was applied to get top sensors using the Boruta package in R [13] and shortlisted top 10 features which were finally used in prediction of engine failures.

3.3 Model Building:-Application of the ensemble classifiers.

a) Machine learning algorithm and R software is used for all experimental results .various experiments were done on Random Forest (Breiman, 2001) classifier, Regularized Greedy Forests, XGBoost with 10-fold cross validation by changing the value of k ranging from 3 to 10 to obtain the best models for prediction of the failures. We used the R language libraries including caret, dplyr, data.table, DMwR, RGF, Boruta, xgboost [13].

b) Evaluation criteria:-Supervised machine learning algorithms are typically evaluated using measures like accuracy, recall, precision.

c) Random Forest algorithm [3] is applied on the data set to predict the time to failure of the engine and parameters optimization and tuning is done using the grid search method to get number of trees with minimum OOB(out of bag error rate) and top 10 parameters are captured from the variable importance plot contributing in the final prediction.

d) XGBoost algorithm [5] is applied on the same data set to predict the time to failure of the engine on the aggregated sensor data set and top 10 parameters are captured from the variable importance plot

e) Finally Regularized greedy forests are applied on the same data set to compare the results and they outperform compared to the other two algorithm for predicting the time to failure of the engine which is shown in the Table 1 of the experimental results.

3.4 Random forest Classifier

Machine-learning algorithm from the "ensemble" family of algorithms which creates multiple models (called weak learners) and combines them to make a decision, in order to increase the prediction accuracy. The main idea of this technique is to build a "forest" of random decision "trees" and use them to classify a new case. Each tree is generated using a random variable subset from the candidate's predictor variables and a random subset of data, generated by means of bootstrap. This algorithm also can be used to estimate variable relevance [10].

3.5 XGBoost

This is an ensemble method that seeks to create a strong classifier (model) based on "weak" classifiers. In this context, weak and strong refer to a measure of how correlated are the learners to the actual target variable. By adding models on top of each other iteratively, the errors of the previous model are corrected by the next predictor, until the training data is accurately predicted or reproduced by the model.

Gradient boosting also comprises an ensemble method that sequentially adds predictors and corrects previous models. However, instead of assigning different weights to the classifiers after every iteration, this method fits the new model to new residuals of the previous prediction and then minimizes the loss when adding the latest prediction. So, in the end, updating of the model happens using gradient descent.

3.6 Regularized Greedy Forests versus Random Forest, XGBoost

In boosting algorithms, each classifier/regressor is trained on data, taking into account the previous classifiers'/regressors' success. After each training step, the weights are redistributed. Miss-classified data increases its weights to emphasize the most difficult cases. In this way, subsequent learners will focus on them during their training [9] [6] [11].

However, the boosting methods simply treat the decision tree base learner as a black box and it does not take advantage of the tree structure itself. In a sense, boosting does a partial corrective step to the model at each iteration.

- In contrast, RGF performs 2 steps:

1. Finds the one step structural change to the current forest to obtain the new forest that minimizes the loss function (e.g. Least squares or log loss)
2. Adjusts the leaf weights for the entire forest to minimize the loss function

- Search for the optimum structure change:

a) For computational efficiency, only 2 types of operations are performed in the search strategy:
   Split an existing leaf node and
   Start a new tree.
b) Search is done with the weights of all the existing leaf nodes fixed, by repeatedly evaluating the maximum loss reduction of all the possible structure changes.
c) It is expensive to search the entire forest (and that is often the case with practical applications). Hence, the search is limited to the most recently-created'‘t’ trees with the default choice of t = 1.

- Weight Optimization

Weights for each of the nodes are also optimized in order to minimize the loss function further:

a) The loss function and the interval of weight optimization can be specified by parameters. Correcting the weights every time 100 (k=100) new leaf nodes are added works well, so this is taken as a default parameter when a RGF model is trained.
b) If ‘k’ is extremely large, it would be similar to doing a single weight update at the end; if ‘k’ is extremely small (e.g., k = 1), it would really slow down the training.

- Regularization

Explicit regularization to the loss function is essential for this algorithm as it overfits really quickly. It is possible to have different L2 regularization parameters for the process of growing a forest and the process of weight correction.

There are three methods of regularization:

a) One is L2 regularization on leaf-only models in which the regularization penalty term

b) G(F) is: $$\lambda \cdot \sum_v \alpha_v^2/2$$

c) The other two are called min-penalty regularizes. Their definition of the regularization penalty term over each tree is in the form of: $$\lambda \cdot \min_{\{\beta_v\}} \left\{ \sum_v \gamma^{d_v} \beta_v^2/2 \;:\; \text{some conditions on } \{\beta_v\} \right\}$$ A larger $\gamma > 1$ penalizes deeper nodes (corresponding to more complex functions) more severely. The degree of regularization may be adjusted through λ or γ hyperparameters.

d) Optionally, it is possible to have different L2 regularization parameters for the process of growing a forest and the process of weight correction

- *Tree Size*

RGF does not require the tree size parameter (e.g., number of trees, max depth) needed in gradient boosted decision trees. With RGF, the size of each tree is automatically determined as a result of minimizing the regularized loss. What we do declare, is the maximum number of leaves in the forest and regularization parameters (L1 and L2).

- *Model Size*

Since RGF performs fully corrective steps on the model/forest, it can train a simpler model as compared to boosting algorithms which require a small learning rate/shrinkage and large number of estimators to produce good results.

IV. PERFORMANCE AND EVALUATION

We tested each of these classifiers to predict on the unknown test data set and compared their Accuracy, Precision and Recall rate. Regularized greedy forest was a clear winner compared to rest of the ensemble classifiers.

Definitions of the evaluation metric used in comparing the performance of three classifiers:-

a) Accuracy = (TruePositive+TrueNegative) / Total

b) Precision = TruePositive / (TruePositive+FalsePositives)

c) Recall = TruePositive / (TruePositive+FalseNegatives)

Ensemble Classifiers	Accuracy	Precision	Recall
Random Forest	77	79	72
XGBoost	81	87	84
Regularized Greedy Forests	84	90	86

Table 1

V. CONCLUSIONS AND FUTURE WORK

The most important conclusion of this work is application of ensemble based technique Regularized Greedy forest and compare its accuracy and performance on Turbo Engine Data set. As predictive maintenance solution in aviation industry is a very crucial.
Second important conclusion is feature engineering and transformation where new aggregation methods were used and advanced mathematical functions were applied on the sensors data which considerably improved the performance of the prediction. We will continue the work in this area, investigating more complex machine learning approaches. Current classification accuracy is great compared to other machine learning algorithms discussed in the paper, but we can further increase it as we get access to more data and as we replace generic algorithms with more specialized ones like regularized greedy forests which are still under research.

VI. REFERENCES

[1] https://www.diva-portal.org/smash/get/diva2:789498/FULLTEXT01.pdf

[2]Ahmed, M., Baqqar, M., Gu, F., Ball, A.D., 2012. Fault detection and diagnosis using principal component analysis of vibration data from a reciprocating compressor, in: Proceedings of the UKACC International Conference on Control, 3-5 September
2012, IEEE Press.

[3] Breiman, L., 2001. Random forests. Machine Learning 45, 5–32.

[4] Guyon, I., Elisseeff, A., 2003. An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182.

[5] https://xgboost.readthedocs.io/en/latest/model.html

[6] Johnson, Rie and Tong Zhang. 2014. "Learning Nonlinear Functions Using Regularized Greedy Forest." IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 36, No. 5

[7] A. Saxena and K. Goebel (2008). "PHM08 Challenge Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

[8] Jardine, A.K., Lin, D., Benefic, D., 2006. A review on machinery diagnostics and Prognostics implementing condition-based maintenance. Mechanical Systems and Signal Processing 20, 1483–1510.

[9]https://www.analyticsvidhya.com/blog/2018/02/introductory-guide-regularized-greedy-forests-rgf-python/

[10] http://www.scielo.br/pdf/spmj/v135n3/1806-9460-spmj-135-03-00234.pdf

[11] Rosset, Saharon, Ji Zhu, and Trevor Hastie. 2004. "Boosting as a Regularized Path to a Maximum Margin Classifier." Journal of Machine Learning Research, Vol. 5, No. 1

[12] https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

[13] https://cran.r-project.org/web/packages/excel.link/index.html