Design and Implementation of Equipment Maintenance Predictive Model Based on Machine Learning

Xiang Li[1], Li Wei[2], Jianfeng He[2]

[1]Jiangxi Engineering Research Center of Process and Equipment for New Energy, East China University of Technology, Nanchang 330013, China
[2]School of Software, East China University of Technology, No.418 Guanglan Avenue, Nanchang Economic and Technological Development Zone, Nanchang 330013, China
Email:tom_lx@126.com

Abstract. Equipment replacement is a fundamental problem in the current industrial Internet of Things application and development and has very practical application significance and value. This paper takes the service life of equipment as the research object, applies machine learning theory, builds the equipment maintenance prediction model based on random forest regression on the basis of analysis and comparison, and gives the typical technology implementation and verification of the model, which has good adaptability and reference value.

1. Introduction

As more and more IoT devices are put into use, equipment maintenance problems also follow, especially for devices in bad environment. It is more urgent to do good maintenance for them, so it can be predicted that their life cycle will greatly reduce the maintenance cost and thus reduce the loss caused by damage. Because the shift in maintenance strategy from so-called post-event control to solving problems by analyzing and enabling predictive maintenance undoubtedly shows us a potential market. Research shows that the development of the Internet of Things and advanced analysis have driven the entire market to adopt a predictive maintenance strategy, resulting in an efficiency increase of 25%-30%. According to the report released by IoT Analytics, the compound annual growth rate (CAGR) of predictive maintenance during 2016-2022 is 39%; in addition, annual technical expenditure will reach $1.096 billion by 2022.

Inside the industrial equipment, there can be dozens of sensors or other health detection data, and the data is organized into a certain format of information, and then evaluated together with maintenance records and machine operation history to determine which problems may occur. There are many companies offering IoT analytics platforms such as GE's Predix platform and Asset Performance Management (APM) suite. It supports the connection of the Internet of Things and the machine, and uses the platform's machine learning algorithm, APM standard measurement and advanced analysis to analyze the data, so that the maintenance staff can timely find out the possible problems of the machine.

Therefore, the use of machine learning algorithms for processing is an important method to solve a large numbers of data analysis, which makes data analysis a more automated process. For example, in some industrial applications, the machine automatically recognizes anomalous performance in production data and allows the machine to automatically set or reconfigure the machine to correct poor production. With the accumulation of machine learning algorithms, this kind of analytical prediction will become an increasingly viable way to improve efficiency. Therefore, the analysis and research of this model has more realistic significance and value.

2. Random Forest Model

For model selection, one principle is usually followed: if it is a discrete variable prediction, choose a classification model; if it is a continuous variable prediction, select the regression model. For example, to predict tomorrow's temperature is a regression task. and to predict whether tomorrow will be sunny or rainy is a classification task. Obviously, it is reasonable to choose the regression model for life prediction here. Linear regression, Poisson regression, decision tree regression, random forest regression, neural networks, etc. are used for regression tasks. Linear regression is simple and fast, but it's very demanding for the data sample, it must be linear, and it's probably a little bit too simple for the problem we are studying. Models that also require linear samples are Poisson regression, Bayesian linear regression, etc., and these models are not very accurate. At the same time, because the neural network has a long set-up time, it has high requirements on the machine configuration, which is not suitable for the current situation.

A random forest is a combined classifier model consisting of decision tree classifier sets $\{h(x,\theta_k), k=1,2,\cdots\}$, where parameter sets $\{\theta_k\}$ are independent and identically distributed random vector and x is the input vector. When given an input vector, each decision tree has one vote to select the optimal classification result. Each decision tree is an unpruned decision tree constructed by a classification regression tree (CART) algorithm. Therefore, corresponding to CART, random forests are also divided into random classification forests and random regression forests. At present, the application of random classification forests is more common, and its final result is the simple majority vote of single tree classification results. The final result of random regression forest is the simple average of the output of a single tree. The random forest uses the Bootstrap resampling technique to repeatedly randomly extract k samples from the original training sample set N to generate a new training set sample set, and then generate a k decision tree based on the self-help samples. The essence is an improvement of the decision tree algorithm, which combines multiple decision trees together. The establishment of each tree depends on an independently extracted sample. Each tree in the forest has the same distribution, and the classification error depends on the classification ability of each tree and the correlation between them.

According to the principle and basic idea of random forest, the generation of random forest mainly includes the following three steps:

First, k training sample sets are extracted from the original sample set S by the Bootstrap method. In general, the sample size of each training set is consistent with that of S;

Secondly, k training sets are learned to generate k decision tree models. In the process of decision tree generation, it is assumed that there are a total of M input variables, and F variables are randomly selected from M variables. Each internal node is divided by the optimal split method on the F characteristic variables, and the F value is constant during the formation of random forest model.

Finally, the results of the k decision trees are combined to form the final result. For the classification problem, the combination method is a simple majority voting method; for the regression problem, the combination method is a simple averaging method.

3. Maintenance Prediction Model Design

This paper chooses a random forest regression model, which performs well on data sets compared to other algorithms. The model is capable of handling very high dimensional data without the need for feature selection. In addition, it can give important features after training is completed. The generalization ability of the model is relatively strong, and the interaction between features can be detected during the training process. Scikit-learn provides an implementation of the model, so you only need to provide the appropriate parameters to create a model. The model parameters are as follows:

n_estimators: number of submodels;

criterion: a calculation method used to determine whether a node continues to split;

max_features: the maximum number of features involved in the judgment when a node is split;

max_depth: the maximum depth, ignored if the max_leaf_nodes parameter is specified;

min_samples_split: the minimum number of samples required for splitting;

min_samples_leaf: the minimum number of samples of leaf nodes;
min_weight_fraction_leaf: the weight value of the smallest sample of leaf nodes;
max_leaf_nodes: the maximum number of leaf nodes;
bootstrap: whether the bootstrap to extract the sample;
n_jobs: parallel numbers, more jobs can speed up the construction of the tree;
warm_start: whether to start hot, if so, the next training will be in the form of an append tree;
oob_score: whether to calculate the outer bag score;
random_state: the randomizer object;
verbose: whether to display the submodel construction log, 0 means no display, 1 means occasional output, and more than 1 means each submodel outputs.

After many tests, the results have not been fitted, and the random forest regression does not need to pay too much attention to its branches and leaves and allow it to grow wildly, Therefore, only the number of sub-models (n_estimators) and the standard criterion for splitting can be concerned here, and verbose can be set if the training progress needs to be observed.

There are two options for criticion:

(1) Mean square error "mse":

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i^{'} - y_i)^2 \tag{1}$$

y' represents the predicted value of the model to the sample, and y represents the true value, and the difference between the two is obtained by subtracting them. The square is a way to get a positive value. You can also take the positive value by taking the absolute value, which is the average absolute value error.

(2) Average absolute error "mae":

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i^{'} - y_i\right| \tag{2}$$

The variables have the same meaning as above, but here the influence of some extreme values on the overall sample data can be reduced, and these extreme values are often data noise.

Introduction to the principle of tree growth: The above discussion of the criterion parameter is for most usage scenarios. The basis for splitting and tree growth in decision tree nodes will be different, but still choose mse as the standard here, as follows:

$$\underbrace{\min}_{A,s}\left[\underbrace{\min}_{c1}\sum_{x_i \in D_1(a,s)}(y_i - c_1)^2 + \underbrace{\min}_{C_2}\sum_{x_i \in D_2(A,s)}(y_i - c_2)^2\right] \tag{3}$$

For the data sets D1 and D2 that are divided into two sides of the arbitrary division point s corresponding to arbitrary partition feature A, find the minimum mean variance of each set of D1 and D2, and the corresponding feature and feature value partition points of the sum of the minimum mean variance of D1 and D2. In addition, c1 is the output mean of the samples of the D1 data set, and c2 is the output mean of the samples of the D2 data set.

The predicted value of each regression tree is the mean value of the leaf nodes of the tree, so the predicted value of the random forest is the average value of the predicted values of all trees.

4. Technical Implementation of the Model

4.1. Model Training

This can be done by using Python and Scikit-learn, and the model can be serialized into a model file by using the sklearn.externals.joblib module:

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import accuracy_score
```

```
from sklearn.externals import joblib
import matplotlib.pyplot as plt
from sample_split import get_samples
if __name__ == '__main__':
    x_train, x_test, y_train, y_test = get_samples()
    model = RandomForestRegressor(n_estimators=200, criterion="mse", verbose=2)
    model.fit(x_train, y_train)
    joblib.dump(model, "aircraft_engine_200.m")
```

4.2. Model Loading

Since different models may be loaded differently, you need to write the appropriate load module for each model. But what's common is that you need to provide a predict method that can directly pass the data for prediction and then return directly to the predicted result. The loading module of this model is as follows:

```
import pandas as pd
from sklearn.externals import joblib
class AircraftEngine200(object):
    def __init__(self):
        self.__model = joblib.load("./aircraft_engine_200.m")
    def predict (self, data):
        result = self. _model.predict(data)
        return result
if __name__ == '__main__':
samples = pd.read_csv("./small_samples_500.csv")
    obj = AircraftEngine()
    del samples["id"]
    del samples["cycle"]
    del samples["RUL"]
    print(obj.predict(samples))
```

The loaded random forest regression model is shown in figure 1, where the number of trees is 100. As the leaf nodes in the model are more divergent, only part of the graph is captured here. Model comparison diagram is the comparison between the model generated by a sample and the optimal model, as shown in figure 2, eventually, the optimal values for all models are produced.

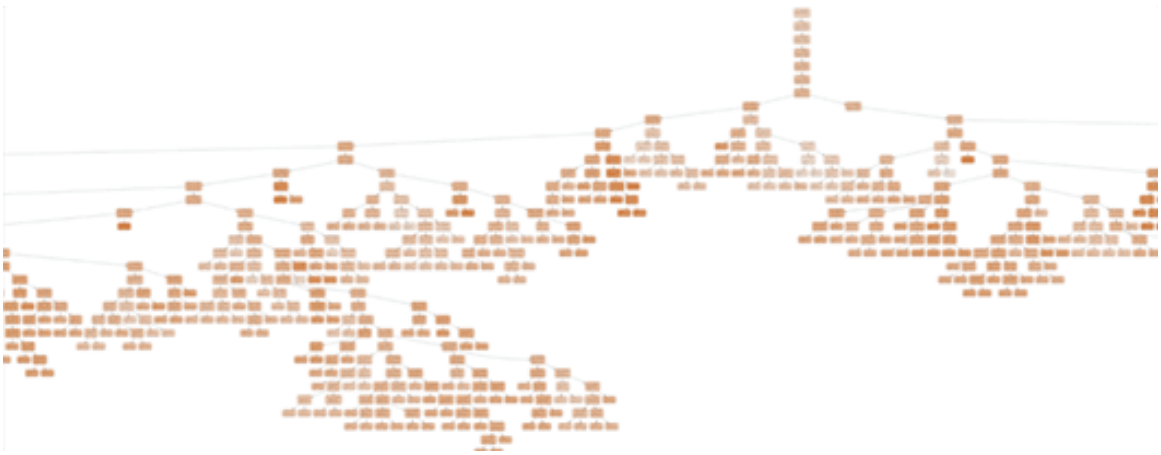

Figure 1. Random forest regression model diagram.

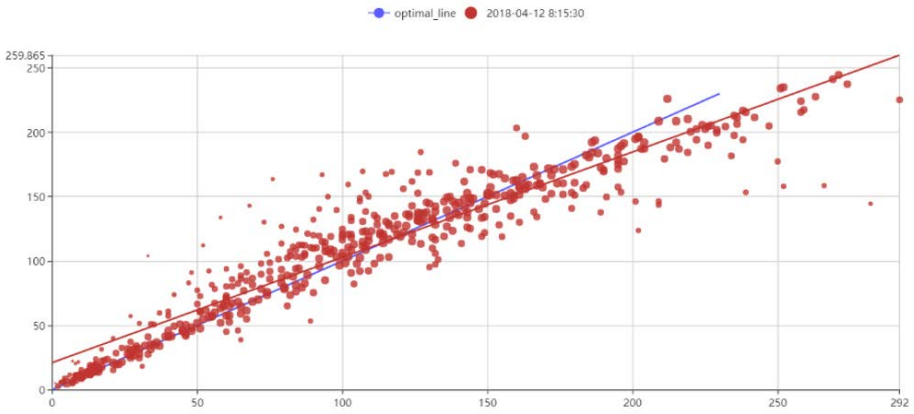

Figure 2. Model comparison diagram.

5. Test and Result Analysis

The "equipment life prediction system" runs under the Ubuntu system. The system operating environment test results are shown in the following table 1.

Table 1. Equipment life prediction system" run test.

test environment	CPU	RAM	Startup mode	Number of models	Test Results	Success probability
environment 1	1 core	1GB	Debug	2	success	100%
environment 2	1 core	2GB	Debug	4	success	80%
environment 3	1 core	1GB	Gunicorn	2	success	100%

Table 2. System unit test.

Function name	**Operation**	**Expected results**	**Results**
Load model interface	Browser direct access	Display system main interface	success
Loading model	Open local model	Show preview	success
Model upload	Click the upload button	Start uploading the model	success
Data prediction interface	Click the data forecast button	Display data prediction interface	success
upload data	Open local file	Load local file and display preview	success
Model viewing interface	Click the model view button	Display model view interface	success
View model	Choose a model	Display model information	success
View log interface	Click to view the log button	Display the view log interface and load the background log	success
Evaluation record interface	Click the evaluation record button	Display the evaluation record interface and load the record	success
View record	Select a record	Draw a chart based on the selected record	success
Record comparison interface	Click the contrast button	Enter record comparison interface	success
Record comparison	Select a record	The chart will be drawn according to another record selected	success

Limited by device RAM, the maximum limit that an expensive 2GB cloud server can boot is four models (affected by model size). But if you use gunicorn to start the server, the pressure will be smaller, because two threads will be started in debug mode, which means that the number of models loaded will be twice as many.

Unit testing is the most basic part of software dynamic testing, and it is also one of the most important parts. The unit testing of this system is shown in Table 2 below.

Each function test of the system is normal, and the functions expected to be realized are completed.

6. Conclusion

This paper combines the actuality of equipment maintenance forecasting work, based on the engine life prediction data, and systematically discusses the construction ideas, processes and Python implementation ideas based on random forest machine learning model.

7. Acknowledgments

This work was by Supported by the National Natural Science Foundation of China (No.41862012 and No.11865002) and the Open Project Program of Jiangxi Engineering Research Center of Process and Equipment for New Energy, East China Institute of Technology. Nos. JXNE2016-10.

8. References

[1] Wang Y H, Deng C, Hu X H, Gao J and Huang C M, *Failure time estimation for mechanical device based on performance degradation*. Computer Integrated Manufacturing Systems.2015,21(08):2147-57.

[2] Zhang B, *Research on Data-Driven Performance Degradation Modelling and Remaining Useful Life Prediction for Mechanical Equipments*. University of Science and Technology Beijing,2016.

[3] Shi H J, *Research on Degradation data driven prediction for equipment residual life*. Taiyuan University of Science and Technology,2015.

[4] Zhang Z X, Hu C H, Si X S and Zhang W, *Degradation Modeling Remaining Useful Life Prediction with Bivariate Time Scale*. Zidonghua Xuebao/Acta Automatica Sinica 2017,43(10):1789-98.

[5] Susto G A, Schirru A, Pampuri S, Mcloone S and Beghi, *A Machine Learning for Predictive Maintenance: A Multiple Classifier Approach*. IEEE Transactions on Industrial Informatics. 2015,11(3):812-20.

[6] Chen W, Xie X S, Wang J L, Pradhan B, Hong H Y, Bui D T, Duan Z and Ma J Q, *A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility*. Catena. 2017,151:147-60.

[7] Li X H, *Using "random forest" for classification and regression*. Chinese Journal of Applied Entomology. 2013(10):1190-97.

[8] Huang Y, Zha W X, *Comparison on Classification Performance between random forests and support vector machine*. Computer Engineering and Software.2012(12):107-10.