Developing a Machine Learning Model to Assign Health Rating to General Equipment Fleet

May 22, 2020

Abstract

3. Proposed framework

The railroad firms are making efforts to change their preventive maintenance methods to predictive maintenance ones. Compared to the preventive maintenance methods, which consist of periodically planned maintenance, the predictive maintenance methods aim at anticipating the time of the failure. In this paper, we develop a machine learning based predictive maintenance framework to assign a health rate score to each railcars of a major railroad company (RC). In this framework, we detect the major roots (components) of railcar failure of RC, and we collect the historical data to build data-driven models for each components. We use the outputs of data-driven models to assign health-rate score to the railcar. We also use health rate scores of railcars as the indicator of future maintenance needs. The health rate score also serves as a decision making tool for maintenance planning, scrap lists and fleet analysis. In this section, We first provide a background for each selected components of a railcar to indicate the impact of failure of components on the failure of the railcar. Then, we introduce the general framework of proposed predictive maintenance method. Finally, we describe each major steps of framework in details.

3.1. Background

Engineer needs to write the general information of railcar and why we chose 4 components as the major roots of railcar failure. Also, describing the functionality of each component and the importance of each (This part needs to be similar to introduction but in summary).

3.2. General framework

The goal of proposed predictive maintenance framework is to detect the railcars that are more probable to fail. This framework assist managers to allocate their resources efficiently to identify the railcars that need maintenance. The framework consists three major steps shown in Figure (1).

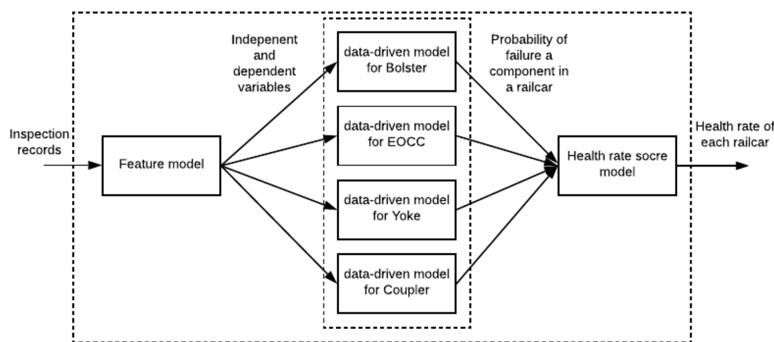


Figure 1: Proposed Framework

The input of this framework is the inspection records for each component of railcars, and the output is the health rate of railcars of RC. In the first step, before constructing a data-driven model for each designated components, it is necessary to generate the feature model. In the feature model, we use a cut-off time to define the independent and dependent variables based on historical failure data of each components. We also engineer some independent features to obtain more information which will be explained in details in future.

In the second step, we develop a data-driven model to classify the failure and non-failure observations for each component. The outputs of data-driven models for each component are the probabilities failure of component in the railcars. We also use two different metrics Area Under

Curve (AUC), and gain chart to evaluate the data-driven models of each components. We will discuss in details why we select these evaluation metrics. If the results from evaluation metrics are satisfactory, we move to the third step. In the third step, we use the outputs of the data-driven models of each component to calculate the health score of each railcar, helping the decision maker to select the proper railcar for the maintenance.

3.3. Feature model

We acquire a dataset from a major railroad company (RC) in United States. This dataset contains the inspection records of four components of railcars. Each inspection records has different features related to the railcars and components. In addition, each records shows that whether a component in the a railcar has been changed or not. This dataset is proper for the preventive maintenance methods. However, in predictive maintenance methods, we need to use historical data to predict whether a component in a railcar is going to fail in following years or not. To do so, as shown in Figure (2), we first split the inspection records based components. Thus, we have a four different datasets with inspection records for each components.

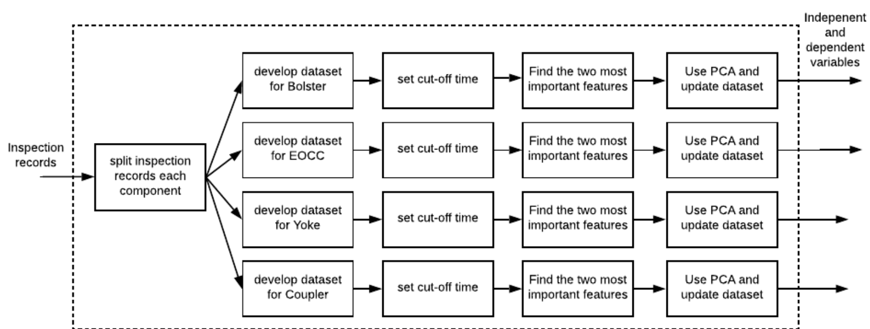


Figure 2: Feature Model

Then, to make sure the data-driven models of each components can properly predict future component failures of railcars, we set a cut-off time. As shown in Figure (3), the cut-off time is January 2019.

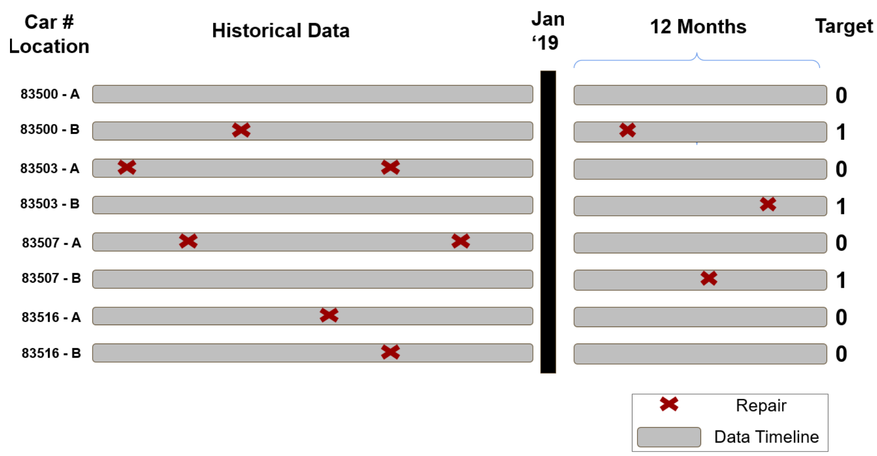


Figure 3: cut-off time to build dependent and independent variables

As seen in Figure (3), each samples of the dataset for a component is the railcar number with a location. The component has two locations, where inspection on the locations of the component happen individually. The dependent variable (target) is that whether a component of a railcar has been failed after January 2019 or not. Let J and I be the number of railcars, and components. We denote y_{ij} as the dependent variable such that

$$y_{ij} = \begin{cases} 1 & \text{if the component i of railcar j failed after the cut-off time,} \\ 0 & \text{Otherwise.} \end{cases}$$

We also require to define new independent features for each railcar based on the inspection records we have for a railcar before the cut-off time. We define nearly 60 different independent features such as, the average number of failure record, component age, and the mileage since last repair. Let $\mathbf{x}_{ij} = [x_{ij1}, ..., x_{ijK}]$ be the dependent features vector of component i of railcar j, where K is the number of dependent features. We categorize these dependent feature based on component features and railcar features. In following Table (1), we show some of these features.

Table 1: Dependent features of model

Component feature	railcar feature
Mileage on Component Since Last Replacement	Car Age
Component Age	Loading Count
Pocket Number	Loading regions
Condition Code	Average days in service

In the next step of feature model, we explore which new independent features have more information to assist us to classify the non-failure and failure events. We use a data-driven model to find the most important features in the dataset. Then, we deploy Principal Component Analysis (PCA) to develop PCA features based on the some important independent features. Let $x_{ij}^{PCA} = [x_{ij1}^{PCA}, ..., x_{ijm}^{PCA}]$ be the PCA features vector of component i of railcar j. Then new vector of independent feature vector of component i of railcar j is $\tilde{\mathbf{x}}_{ij} = [x_{ij1}, ..., x_{ijK}, x_{ij}^{PCA}]$.

3.4. Data-driven model

The data-driven model is a mathematical function that estimates as close as possible to the actual true function of interest. Let $\hat{f}_i$ be data-driven model of component $i = 1, .., I$, where $\tilde{\mathbf{x}}_i = [\tilde{\mathbf{x}}_{i1}, ..., \tilde{\mathbf{x}}_{iJ}]$ is independent feature variables matrix , and $\hat{\mathbf{y}}_i$ are the predicted outcome of the data-driven model of component i.

$$\hat{f}_i : \tilde{\mathbf{x}}_i \rightarrow \hat{\mathbf{y}}_i \ \forall i = 1, .., I$$

For each component i, we build a data driven model. The input of the model is the information of the railcar before the cut-off time $\tilde{\mathbf{x}}_i$. The goals of the data driven models for a component is to predict whether the component of a railcar will fail after the designated cut-off time. We also obtain the probability of the failure of component i in car number j, p_{ij} from data driven models. We randomly separate the dataset of component i into train and test sets. In training process, we use a set of railcars where randomly selected to train the model. To test model, we use the remaining railcars to validate our data-driven model for the component.

3.5. Evaluation metrics

The goal of this part is to evaluate the selected data-driven models and tune their hyper-parameters. We use two different metrics which are AUC and gain chart. These two metrics are evaluating the models in different perspectives. Gain chart is a visual presentation to show that how many of failures can be inspected within selected samples compared to the random selection. This metric is used when companies face a limited budget for a inspection and they want to select only a fraction of the population for the inspection.

Another metric we use to evaluate the data-driven models is AUC which is a performance measurement for classification problem at various thresholds settings. AUC provides an aggregate measure of performance across all possible classification thresholds. It tells how much model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. We use this metric because we have imbalanced data. The number of recorded failure in each components is smaller than the number of recorded non-failure. In a component, a only 0.02 of records have failures. It means that the component never failed in most of railcars. Therefore, it is important to balance between false positive and false negative in the confusion matrix. To balance between these two values for a given hyper-parameters of the data-driven model, we use threshold as a decision variable. False positive is the number of the failure records in the test set where predicted as the non-failure records. False negative is the number of the non-failure records in the test set where predicted as the failure records. This two values are important because repairing a healthy railcar is useless, and not predicting a failure will cost RC.

To balance between these false positive and false negative for a given hyper-parameters of the data-driven model, we set a threshold as a decision variable. Let t be threshold The objective is to maximize the AUC of a given data-driven model $f(.)$.

$$AUC(f) = \max_t AUC(t, f)$$

We denote $AUC(t, f)$ as AUC function of the data driven model f with threshold t. We use above optimization problem to find maximum AUC for a given f, defined as $AUC(f)$.

3.6. Heath rate score model

In the previous steps, we develop the data-driven models for each component independently. However, we need to figure out which railcar need to be inspected. In order to do that, we need to assign a health-rate score to each railcar. The probability of the failure for each component, derived in the previous steps, will be utilized to calculate the final health rate for each railcar. Let p_{ij} be the probability of the failure of component i in railcar number j which is the output of the data-driven model of component i. We denote w_i as the component critically associated with component i. Therefore, the health rate (HR) for each car number j is defined as follows (i corresponds to the components)

$$HR_j = \sum_i w_i p_{ij}.$$

The process has been shown in Figure (4).

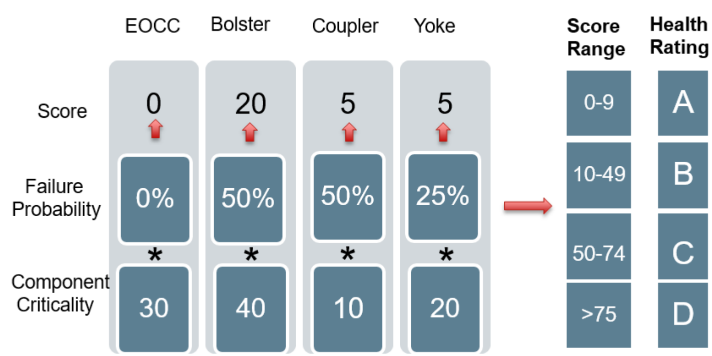


Figure 4: Health rate score model

We used the engineering team's feedback to assign weights to each these components and the following weights will be used to generate the final health rate. We first put the priority for inspection on the railcars with health-score rate D. Figure (5) shows the predictive maintenance framework to assign health rate score to each railcar.

Algorithm 1 Proposed predictive maintenance framework

Step 1. Set the cut-off time to define independent variables matrix $\mathbf{x}_i$ and the target vector $\mathbf{y}_i$.

Step 2. Find the important feature and use PCA to add a new independent variable. Update independent variables matrix $\tilde{\mathbf{x}}_i = [\mathbf{x}_i, \mathbf{x}_i^{PCA}]$.

Step 3. Separate the dataset of each component to train and test sets. Train each data-driven model $\hat{f}_i$ $\forall i = 1, .., I$. Calculate p_{ij}.

4. Calculate the health rate score of each car number j which is $HR_j = \sum_i w_i p_{ij}$.

Figure 5: Pseudo-code of the proposed predictive maintenance framework

4. Numerical Study

In this section, we use the data from a well-known railroad company (RC) in USA to validate the proposed predictive maintenance framework. First, we explain the dataset in detail. Then, we analyze the initial features of each record in the dataset to engineer the new features via feature model. We select Random Forest Classifier (RFC) as our primary data-driven model, and we evaluate the model based on Gain Chart and AUC. Finally, we used Heath rate score model to rank the railcar based on their health.

4.1. RC Dataset

In this section, we first identify the major reasons of railcar failure which correspond to failures of EOCC, bolster, yoke, and coupler. For each of these component, we have a dataset of inspection records from 1984 to 2020. Each dataset contains the inspection records of more than 10,000 railcars. Before we use these datasets to develop dependent features to build the data-driven models, we conduct a preprocessing operation. We need to remove the class I inspection records. In the class I inspection, the repair stations change all components of railcar without inspecting a failure in any components. This is important to remove these inspection records because, no failure did not happen, and considering those records can mislead the data-driven model to predict the component failure.

4.2. Feature model

As we discussed in section 3, we need to engineer features to predict whether a component of a railcar will fail in future or not. To do so, we split inspection record based on component. We want to know based on the historical data of a component, the component of a railcar will fail in the next following year. Therefore, we set the cut-off time which is January 2019. For the dataset of a component, if the component of a railcar fail after January 2019, the dependent feature of the railcar will be 1 meaning failure; otherwise 0 which is non-failure. We develop several independent features which can be categorized as the railcar features and the component features which were calculated based on the inspection record of the component before January 2019. We showed some of these feature in Table (1).

We first noticed that we have imbalanced datasets. Table (2) shows the ratio of failure to whole samples in each dataset.

Table 2: Imbalance dataset

Component	EOCC	Coupler	Yoke	Bolster
Ratio of number of failures to whole samples	0.033	0.034	0.017	0.003

The above table indicates that we are facing highly imbalance dataset, especially in Bolster. We also use the random forest classifier to identify the important features in the datasets of different components which has been shown in Table (3).

Table 3: Important features in different data set

EOCC	Yoke	Coupler	Bolster
component age	component age	component age	mileage since last replacement
car age	average day of trip being loaded	mileage since last replacement	average day of trip being empty
mileage since last replacement	car age	car age	car mileage
car mileage	mileage since last replacement	average day of trip being empty	component age
average day of trip being loaded	car mileage	car mileage	car age

It is interesting to notice that the four features component age, car age, mileage since last replacement, and car mileage are one of five important features in all datasets. In addition, component age and mileage since last replacement have correlation nearly equal to 0.95 in all datasets. car age and car mileage has nearly correlation 0.87 in all datasets. Therefore, we build random forest classifier model without correlated counterparts. We observe that in the model without component age, mileage since last failure is one of the most important features. In addition, in the model without mileage since last failure, component age is the one of the most important features. This also true for car age and car mileage. Then, we use PCA to create two PCA features based on mileage since last failure and component age, and car age and car mileage. we add these PCA features to the datasets.

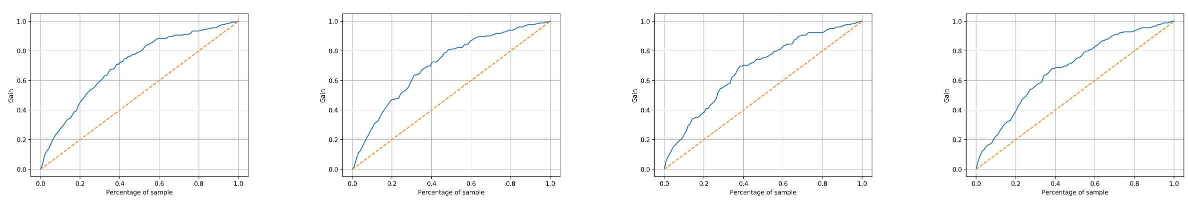
4.3. Data-driven model and evaluation

The goal of our data driven model for a component is to predict whether the component of a railcar will fail after the designated cut-off time or not. For each component, we build a data driven model. The input of the model is the engineered features of the railcars before the cut-off time. We select Random Forest for all different components as the data-driven model to predict the component failure of railcar after cut-off time. We choose random forest classifier because we have categorical and quantitative features. To examine the effect of including PCA features, we compare the AUC of B-PCA-K datasets (dataset with PCA features with keeping car age, car mileage, component age, and mileage since last failure) with AUC of W-PCA dataset (dataset without PCA features). We evaluate AUC of B-PCA-NK dataset (dataset with PCA feature without keeping car age, car mileage, component age, and mileage since last failure). We also analyze the over-sampling method using Adaptive Synthetic (ADASYN) without PCA, and ADASYN with PCA (ADASYN-PCA). The results have been shown in Table (4).

Table 4: 10-fold AUC of different datasets

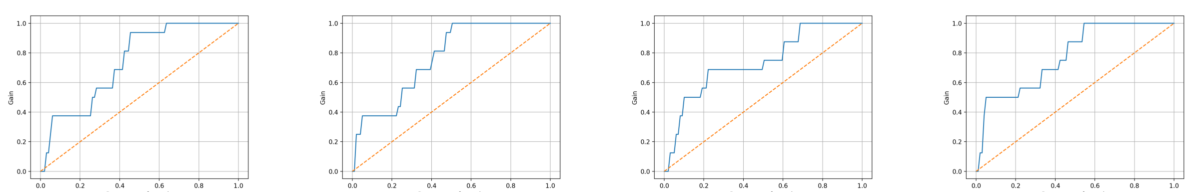
dataset	EOCC	Yoke	Coupler	Bolster
Without-PCA	0.65	0.59	0.58	0.70
B-PCA-K	0.67	0.61	0.61	0.72
B-PCA-NK	0.61	0.57	0.59	0.63
ADASYN	0.63	0.62	0.57	0.73
ADASYN-PCA	0.63	0.62	0.57	0.72

From Table (4), we notice that B-PCA-K AUC in all datasets are greater that datasets without PCA features. In addition, from results of B-PCA-NK, we understand that we need to keep car age, car mileage, component age, and mileage since last failure. We also observe that using both PCA and oversampling (ADASYN) improve AUC. We then deploy gain chart to study the data-driven models with different settings such as Without-PCA, B-PCA-K, ADASYN-PCA shown in Figures (6) to (9).



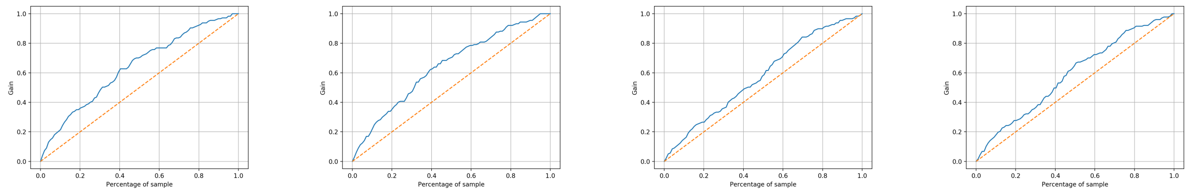
(a) Without-PCA (b) B-PCA-K (c) ADASYN (d) ADASYN-PCA

Figure 6: Gain Chart for EOCC



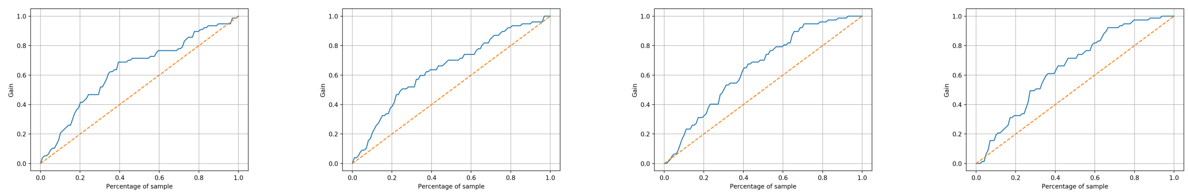
(a) Without-PCA (b) B-PCA-K (c) ADASYN (d) ADASYN-PCA

Figure 7: Gain Chart for Bolster



(a) Without-PCA (b) B-PCA-K (c) ADASYN (d) ADASYN-PCA

Figure 8: Gain Chart for Coupler



(a) Without-PCA (b) B-PCA-K (c) ADASYN (d) ADASYN-PCA

Figure 9: Gain Chart for Coupler

These figures show how many of failures can be inspected within selected samples compared to the random selection. For instance, in Table (5), we show if the firm RC selects 10 percent of sample, how many of failure can she detects.

Table 5: Percentage failure can be detected from 10 percent of sample

dataset	EOCC	Yoke	Coupler	Bolster
Without-PCA	27	21	21	37
B-PCA-K	29	21	23	37
ADASYN	22	19	18	50
ADASYN-PCA	23	19	15	50

For instance, from Table (5), in bolster dataset, with combination of PCA and ADASYN, we can detect 50 percent of failure within 10 percent of samples.

4.4. Health rate score model

I don't know what kind of result we need to put there

4.5. Comparing with other methods for failure detection

It is good to have this with what really company is doing to see the differences

5. Conclusion and future research

blah blah blah

Reference