Combining Boosted Trees with Metafeature Engineering for Predictive Maintenance

Conference Paper *in* Lecture Notes in Computer Science · October 2016

DOI: 10.1007/978-3-319-46349-0_35

Combining Boosted Trees with Metafeature Engineering for Predictive Maintenance

Vítor Cerqueira(✉), Fábio Pinto, Claudio Sá, and Carlos Soares

INESC TEC, Universidade do Porto,
Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal
{vmac,claudio.r.sa}@inesctec.pt, fhpinto@inescporto.pt, csoares@fe.up.pt

Abstract. We describe a data mining workflow for predictive maintenance of the Air Pressure System in heavy trucks. Our approach is composed by four steps: (i) a filter that excludes a subset of features and examples based on the number of missing values (ii) a metafeatures engineering procedure used to create a meta-level features set with the goal of increasing the information on the original data; (iii) a biased sampling method to deal with the class imbalance problem; and (iv) boosted trees to learn the target concept. Results show that the metafeatures engineering and the biased sampling method are critical for improving the performance of the classifier.

Keywords: Predictive maintenance · Anomaly detection · Boosting · Metalearning

1 Introduction

This paper describes a data mining workflow for predictive maintenance of heavy trucks. This type of vehicles are typically operated in a daily basis and used for large trips. They are an important tool in several industrial sectors such as transportation or construction. In this context, it is of fundamental importance that all components comprising these vehicles are regularly maintained. A well done maintenance is key to avoid undesired breakdowns, which can be costly to the company operating these vehicles.

One of those components is the Air Pressure System (APS). The APS generates pressurised air that is used for different tasks in a truck, such as braking and gear changing, making it a core component for maintenance purposes.

The data collected describes several components from heavy Scania trucks in everyday usage. Moreover, in order to guarantee the quality of the predictive model, the data has been sampled from all available data by experts.

In the Data Mining terminology this problem is presented as a binary classification problem, where the positive class of the target concept consists of failures for a specific component of the APS. The negative class consists of trucks with failures not related to the APS. The exploratory analysis of the data enabled to outlined two important conditions: (1) high quantity of missing values and

H. Boström et al. (Eds.): IDA 2016, LNCS 9897, pp. 393–397, 2016.
DOI: 10.1007/978-3-319-46349-0_35

(2) high imbalance in the class distribution. These characteristics of the data raises challenges from the data scientist perspective.

For dealing with (1), we use a filter that excludes the features and examples that present an higher percentage of missing values. This not only reduced the size of the data but also enabled to remove some of the noisy features that were part of the original data.

For dealing with (2), we use SMOTE [1], an over-sampling technique that creates synthetic examples of the minority class. SMOTE enables to balance the class distribution of the data which leads to a better generalization of the classifier.

On top of these issues, the data is completely anonymized for proprietary reasons. This is particularly cumbersome in the feature engineering step. We choose to generate new features by taking a metafeature engineering approach that does not require knowledge about the domain. Most of these metafeatures were generated using unsupervised techniques for anomaly detection problems.

As for the modeling step, we use an ensemble of boosted trees. Ensemble learning has shown to be a good solution in variety of data mining tasks and we also verified a great improvement in the performance of our system by using ensemble approaches. The workflow is summarised in Fig. 1.

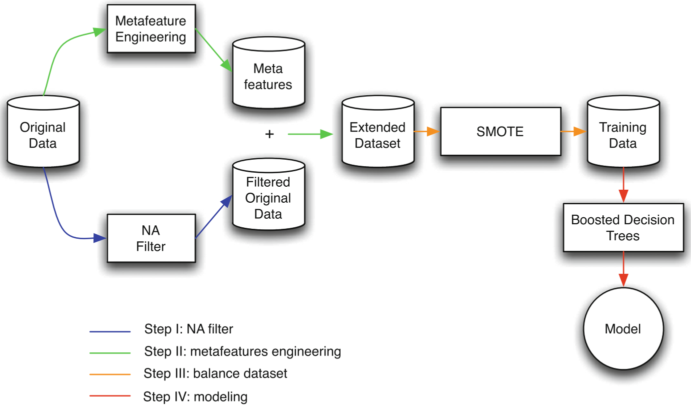

Fig. 1. Schema of our workflow. It starts by dealing with the missing values (Step I). In Step II the metafeatures are computed, which are then added to the original, after applying the missing values filter. Then, in Step III, we use SMOTE to balance the dataset and finally in Step IV we generate our ensemble of boosted decision trees.

The paper is structured as follows. In Sect. 2, we present our approach to deal with the missing values in the data. Section 3 explains the metafeature engineering step that we followed. In Sect. 4 we detail the modeling approach that we used to learn our final model and finally, in Sect. 5, we conclude the paper by discussing the results obtained and the lessons learned from this challenge.

2 Dealing with Missing Values

Dealing with missing values has been widely studied in the literature [2]. The usual techniques, such as listwise deletion, pairwise deletion, indicator variable, and mean substitution could have been an option. However, for simplicity and in this particular case, we decided to remove features with the greater amounts of missing values. It is rather an ad-hoc approach, but there is also no clear consensus in the literature on which approach is better [2].

Some features presented a great number of missing values, up to 80 % in the most extreme case. As an indicator, 8 out of the 170 independent variables, had more than 50 % of missing values.

While paying special attention to the performance of the models, we tested how much features we could remove without affecting the accuracy. We also realized that after removing the features with the most missing values, there was quite a number of duplicates in the data. This seems to indicate that the removed features have little effect on the target.

3 Metafeature Engineering

From the statistical point of view, anomalies are associated with observations with high deviation from the typical behaviour, i.e., outliers. Since the positive class of the data is characterized by rare events in the domain (malfunctions in the APS), we can regard this problem as an anomaly detection one.

In this context, we perform an outlier analysis to the data in order to assess how far each observation is from the norm. The results from this outlier analysis are then embedded as attributes in our data. By doing this meta-level analysis we aim at increasing the information related to the outlyingness of each observation. Therefore, we generated metafeatures using three different outlier detection techniques that we present in the following subsections.

3.1 Boxplot Analysis

The Box plot is a typical way of describing the distribution of some data through some summary statistics (e.g. median, Inter-Quartile Range (IQR)).

Since all the attributes of our data are numeric, we can perform this analysis to each predictor separately. For each attribute we compare each value with the typical value of that same attribute. Then, as explained in [3], if the difference between these two values is high that might be an indicator that something is not right and the respective observation might be an outlier. Furthermore, the size of this difference can be regarded as a measure of outlyingness.

One drawback of this analysis is that it is a uni-variate approach. By analyzing only one attribute at a time, we lose potentially useful global information. To overcome this issue we approach the problem using the Local Outlier Factor (LOF) method, which measures the degree of outlyingness of observations as a whole.

3.2 LOF

LOF [4] is a method for quantifying the outlyingness of an observation by comparing it to its local neighbourhood through density estimation. Essentially, an observation with a very low density has greater probability of being an outlier.

3.3 Clustering-Based Outlier Ranking

The third method we used to analyze the outliers in the dataset is based on an hierarchical Agglomerative Clustering algorithm [5].

Hierarchical Agglomerative Clustering starts with Z groups (Z being the number of observations), each initially containing one object, and then at each step it merges the two most similar groups until there is only one single group, containing all data.

The rationale for this method is that the last observation that are merged might still be significantly different from the group they are merged into. By definition outliers are different cases and will typically not fit well into a cluster, unless that cluster is comprised by other outliers itself. Yet again, since these are not ordinary data points, we do not expect them to form large groups.

4 Modeling

In this section we detail the modeling approach that we used for this challenge.

The model was generated using the XGBoost library [6]. The parameter tuning of the learning algorithm was done using 10-fold cross validation. We payed particular attention to parameter setting in order to avoid overfitting. Given the experimental results that we gathered, overfitting can be a pitfall in this challenge.

The performance of the modeling algorithms should be measured according to a cost sensitive metric defined by the challenge organizes. The intuition for this is because the two error types (false negative - FN and false positive - FP) do not have the same meaning. Sending a vehicle for an unnecessary maintenance (FP) is clearly less costly than facing an unexpected breakdown (FN). For this reason the evaluation metric is computed as follows: $\text{Cost} = \text{FP} \times 10 + \text{FN} \times 500$.

5 Conclusions

Table 1 presents the results for three workflows that we tested. We concluded from the several experiments we carried out that XGBoost seems to be a good option for this problem, particularly when the metafeatures are available for learning. However, we did encounter some issues regarding the tuning of the algorithm in terms of overfitting. For this particular data, the algorithm showed high sensitivity regarding its parameters.

Overall, we think that SMOTE and the metafeature engineering are the most important steps in our proposal.

Table 1. Results estimated using 10-fold cross validation for four methods. A Random Forest with and without metafeatures and the XGBoost algorithm with and without metafeatures. The XGBoost with metafeatures shows the minimum average cost and with the lowest deviance.

Algorithm	Average cost	SD cost
RF without metafeatures	4721	882
RF with metafeatures	4440	900
XGBoost without metafeatures	4030	910
XGBoost with metafeatures	3750	810

Acknowledgements. This research has received funding from the ECSEL Joint Undertaking, the framework programme for research and innovation horizon 2020 (2014–2020) under grant agreement n 662189-MANTIS-2014-1. It was also financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013.

References

1. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
2. Acock, A.C.: Working with missing values. J. Marriage Fam. **67**(4), 1012–1028 (2005)
3. Torgo, L.: Data Mining with R: Learning with Case Studies, 1st edn. Chapman & Hall/CRC, Boca Raton (2010)
4. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data, pp. 93–104 (2000)
5. Torgo, L.: Resource-bounded fraud detection. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS, vol. 4874, pp. 449–460. Springer, Heidelberg (2007). doi:10.1007/978-3-540-77002-2_38
6. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. arXiv:1603.02754 (2016)