Capstone Proposal

Predicting Device Failures with Ensemble Models

Michael Lin

July 2019

1 Domain Background

Modern industrial economies run on machines. From machineries seen on production floors, to tractors used in farms, from cars running on highways, to internet of things deployed in the field, machines are everywhere.

Overtime, machines do wear out, however. The ability to identify which machines become faulty before they actually break down can pay great dividends. A joint study by Wall Street Journal and Emerson found that unplanned downtime costs industrial manufacturers an estimated $50 Billion per year. Equipment failure is the cause of 42% of this unplanned downtime[1].

Taking preemptive action before device failure, or predictive maintenance, is an area of active investigation by practitioners of machine learning and AI.

The use case for predictive maintenance is wide-spread[2]. In automotive, data from sensors in vehicle helps alert drivers of issues that need servicing. In utility, data from smart meters helps detect early signs of issues on the grid. In internet of things scenario, data from sensors in factories and products in combination with algorithms help uncover warnings of costly failures before they occur. And in insurance, data helps predict likelihood of extreme weather conditions.

On modeling front, predictive maintenance does present a unique challenge, namely that of imbalanced data distribution[3]. Failure cases are considered rare events, as most of the time machines function properly. The ability to discern truly failure cases in the training data is what motivates me to select this topic for capstone.

Moreover, we may require more sophisticated modeling approach to pick up the underlying complex relationship between sensor reading and machine failure. Ensemble models[4], combining feedback from many models to yield more accurate prediction is desirable in predictive maintenance. In such case, ability to accurately predict and take preemptive actions may be more important than to have a clear explainability of the cause and effect.

2 Problem Statement

The objective of the capstone is to accurately predict devices that require maintenance, while at the same time, minimize false positive cases (predicted failure cases that are not faulty) and false negative cases (predicted non-failure cases that are actually faulty). Wrongful classification of either type is not desirable. False positives mean unnecessary down time when we take devices out of service rotation, and incur cost of applying maintenance. False negatives translate to liability when machines break down unexpectedly. If the devices are transportation vehicles, people's lives may be at stake. If the devices are critical machineries used in production floors, failures may cause collateral damage in other machineries.

3 Datasets and Inputs

The dataset for the capstone is sourced from a github repository (https://github.com/dsdaveh/device-failure-analysis/blob/master/device_failure.csv).

The fields in the datasets include the following.

- Date
- Device
- Failure Label
- Attribute1
- Attribute2
- Attribute3
- Attribute4
- Attribute5
- Attribute6
- Attribute7
- Attribute8
- Attribute9

The name of the sensor readings are concealed from us. We will not be able to leverage understanding of the sensor types to gauge its impact on device failures. We will solely leverage the power of our model to achieve the objective.

Initial analysis of the dataset reveals the following:

- The dataset is telemetry logs of 124,494 records, collected from 1,169 devices.
- The device failure rate is 9.07%.

- The data possibly comes from 3 different manufacturers (or device types).
- The data spans period of 2015/1/1 to 2015/11/2.
- The duration of device in service varies widely, ranging from 1 day to 304 days.
- For features, we have 9 attributes which are numerical in nature.

4 Solution Statement

We envision the assignment as classification challenge. We use devices' sensor reading to predict whether a particular device will be faulty the next working day. We further propose a constraint to correctly identify as many true failure and non-failure cass as possible, while at the same time, minimize incidences of false positives and false negatives.

In designing the solution, we keep the following considerations in mind:

- Devices do not have equal number of log records. We have to reconstruct a dataset fixing granularity at the device level, with proper labelling of failure or non failure at each device's last point of observation.
- The distribution of the label is highly skewed. We will need case weighting or resampling to ensure our model have sufficient exposure to both failure and non-failure classes during the training process.
- In terms of attributes, we will use sensor reading of the previous day, the previous 2 days, and previous 3 days as predictor. If we use the reading of last day to predict possible device failure, the setup leaves us with no lead time for possible intervention, and won't qualify as a practical model.

5 Benchmark Model

We purposely set aside 20% of data as hold-out. We use this pool to validate our model performance. The remaining 80% of data is for training of model parameters.

And for each round of model training, we propose over-sampling of minority class (failure case) to keep percentage of failure vs. non-failure classes more less fixed throughout the cycle.

In addition, we fix random seeding to ensure results are reproducible.

As for actual model training, we propose a phased approach, starting from building a naive baseline to constructing more sophisticated ensembles.

- Model 1 uses majority class (no failure) to predict validation dataset
- Model 2 fits a decision tree model
- Model 3 fits a KNN model with feature standardization
- Model 4 fits a SVM model with feature standardization
- Model 5 to 8 fit ensemble variants, ranging from AdaBoost, Gradient Boost, Random Forest, to Extra Trees

As final calibration, we investigate whether adjusting prediction cut-off can further improve model performance.

6 Evaluation Metrics

We use area under the curve (AUC) as the scoring criteria to guide our model search, as we move across the folds of the training dataset to estimate the model parameters.

We also use confusion matrix to further provide diagnostics of how well the model is performing in validation, balancing cases of false positives vs. false negatives.

7 Project Design

In terms of actual steps, we will stick to the following analysis flow.

1. clean data, deleting duplicate columns and rows
2. generate features and construct working dataset
3. split dataset into training and validation
4. resample minority class in training dataset
5. establish prediction baseline
6. fit KNN and SVM models for further comparison
7. fit variants of ensemble models for enhanced performance
8. tune hyperparameters
9. adjust prediction cut-off point for last mile improvement
10. discuss other means for continuous improvement

8 References

[1] IndustryWeek, “How Manufacturers Achieve Top Quartile Performance”
https://partners.wsj.com/emerson/unlocking-performance/how-manufacturers-can-achieve-top-quartile-performance/

[2] William Trotman, “5 Use Cases for Predictive Maintenance and Big Data”
https://blogs.oracle.com/bigdata/predictive-maintenance-big-data-use-cases

[3] John Brennan, “Dealing with Imbalanced Data’
https://www.migarage.ai/intelligence/imbalanced-data/

[4] Joseph Rocca, “Ensemble methods: bagging, boosting and stacking”
https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205