Predictive Condition Monitoring

Unplanned downtime is the bane of many companies, costing vast sums in lost profit and customer satisfaction. Planned maintenance is important, but all too often planned maintenance fails to pick up a progressively degrading component, and failure occurs in spite of strict adherence to the maintenance schedule.

Condition monitoring is one part of the solution: sensors capture critical data, and relay it to a database where is is stored and analysed, with the goal of identifying a degrading component, and replacing it, before breakdown occurs.

There are many ways that condition monitoring can be implemented, but in general they can be divided into the following categories

- open source
- outsource
- consultants

Personally I'm a fan of open-source, and in-housing the development and deployment of condition monitoring systems. Most equipment these days is equipped with PLCs which already record critical data. The missing links are establishing pipelines to aggregate that data in a central database, and the analytic tools to quickly and easily identify a degrading component before it fails.

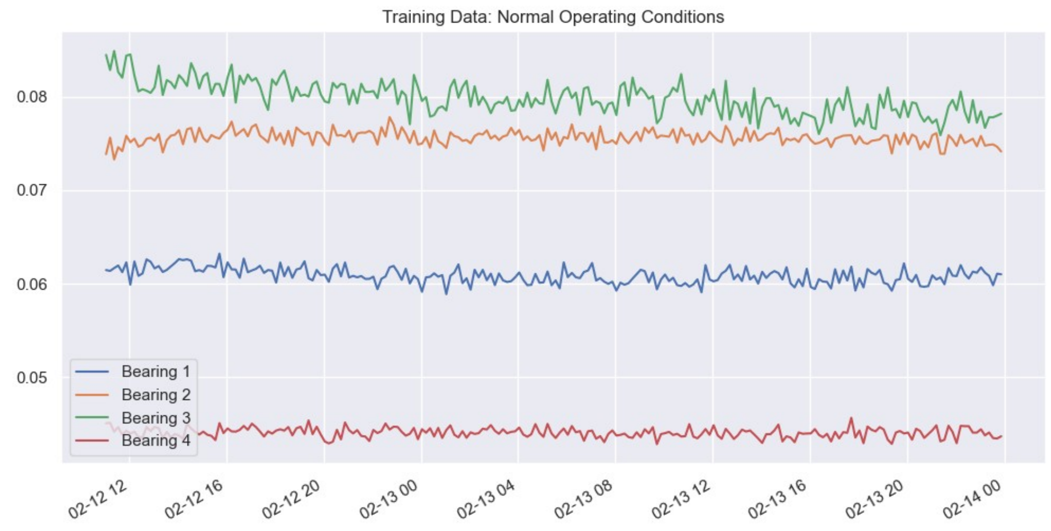
For the cases where existing PLCs don't capture the data required, buying and installing appropriate sensors is relatively inexpensive.

The analytic side is also relatively straight forward, with a thriving community of open source developers who freely share their code.

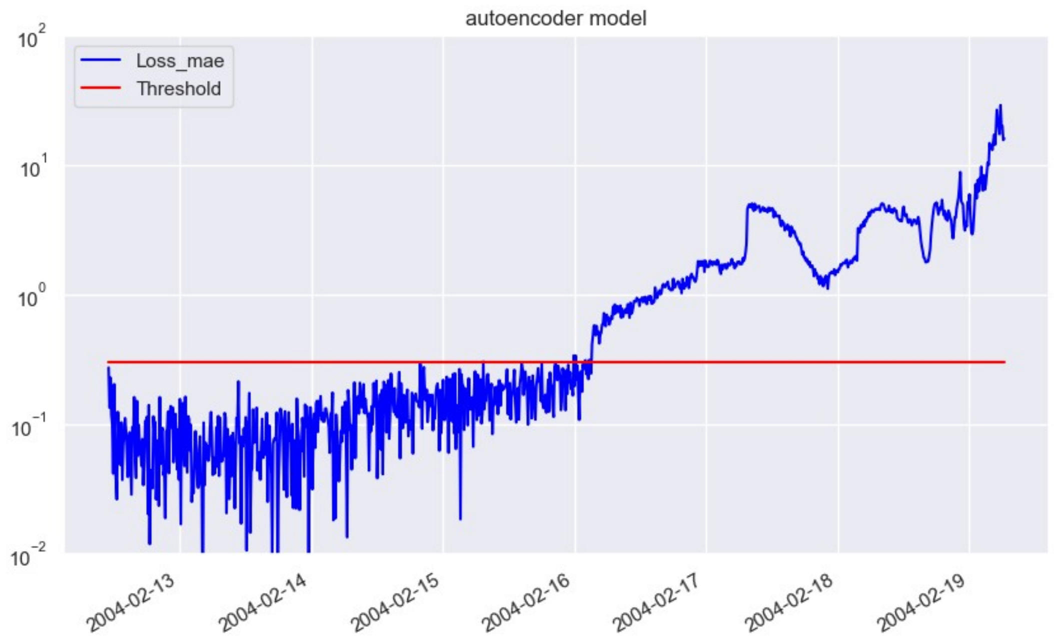
The results of successfully deploying condition monitoring can be remarkable, with unplanned downtime reduced to <1%. As long as you adhere to engineering first principles, this is something that any company with motivation and tenacity can achieve in-house.

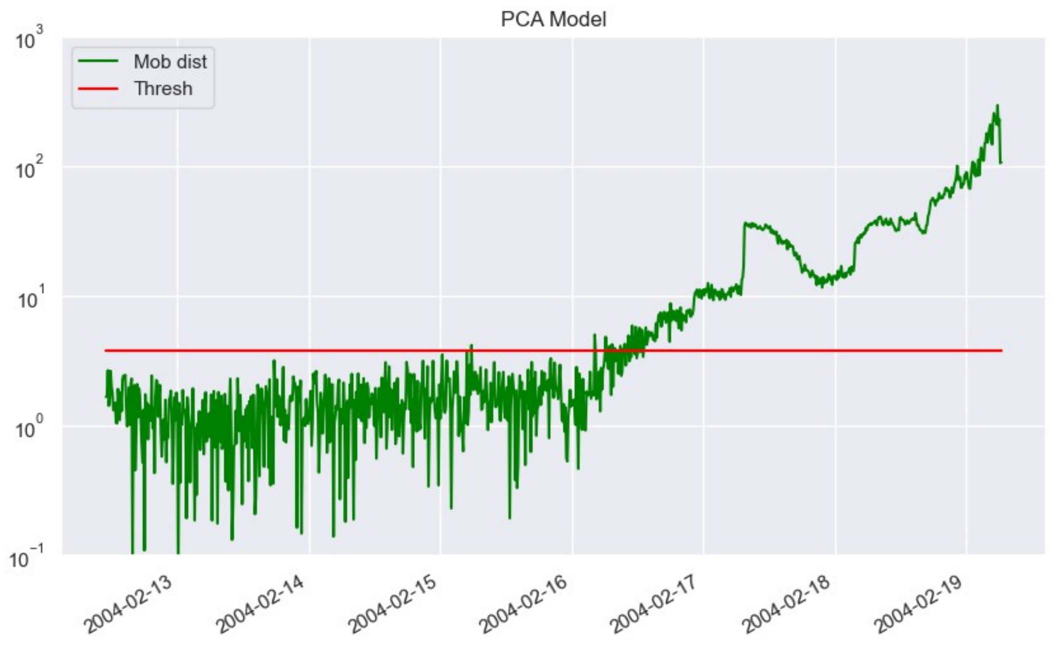
I thought it might provide some insights if I were to share an example that uses deep learning and machine learning to monitor vibration in bearings, looking for early signs of degradation.

We can see from a simple plot that the raw data doesn't really give us a lot of insights



However with a little code, things start to become really clear. I have used two models: an autoencoder model, and PCA (Principle Component analysis). The outputs are shown below.





Both Models tell a remarkably similar story: the bearing will hit the critical threshold on about the 16th, and will fail catastrophically on the 19th. If engineering were to get this information on the 16th, and act immediately to replace the bearings, a breakdown can be avoided.

For the sake of brevity I haven't gone into the details of how we determine the Mahalanobolis Index and distance, or how we determine the critical threshold. That information, the code and data are all shared on GitHub. This little project uses publicly available data, and shows just how easy and effective this can be: bearing condition monitoring. It could be easily turned into a dashboard, so that engineering and maintenance staff can see live data showing the condition of critical equipment. There is no limit to how many components can be monitored.

A key take away point that I'd like readers to understand is that this is not something that you need to hire in consultants for, or pay exorbitant fees to buy an off-the-shelf solution. With suitable motivation, a willingness to embrace first principles and be a little creative, deploying predictive condition monitoring can be done by anyone. You engineering team already has the most important skills. The rest can be learned from publicly available material.

Thanks for reading!