Anomaly Detection for Early Failure Identification on Automotive Field Data

Aditya Jain[1] and Piyush Tarey[2]

[1,2]*Tata Consultancy Services Ltd, Pune, Maharashtra, 411057, India*

aditya.jain7@tcs.com
piyush.tarey@tcs.com

ABSTRACT

The automotive industry is witnessing its next phase of transformation. The vehicles are getting defined by software, becoming intelligent, connected and more complex to design, develop and analyze. For these complex vehicles, prognostics and proactive maintenance has become ever more critical than before.

OEMs and suppliers analyze probable failures that a vehicle component is likely to encounter, define fault codes to identify those failures, and provide procedure or guided steps to resolve them. For smarter vehicles, it is required that vehicles be capable to catch potential problems as soon as the component's condition starts to deteriorate and becomes a failure. These failures could be known (defined) or new (undefined). Given the vehicle development timelines and increasing complexity, many problems are not analyzed at design stage and remain undetected before production. Hence, no fault code or test case exist for them. Diagnosing such problems become very difficult, postproduction.

The aim of this paper is to propose a Machine Learning (ML) based framework which utilizes minimally labelled or unlabeled sensor data generated from a vehicle system at a given frequency. The framework utilizes an ML model to identify any anomalous behavior or aberration, and flag it for further review. This framework can be adopted on large amount of real time or time series data to identify known as well as undefined failures early. These models could be deployed on cloud or on edge (on vehicles) for analyzing real-time sensor data for a given system/component and flag any anomaly. It could further be utilized to create a part specific Predictive Maintenance (PM) model to provide proactive warnings and prevent downtime.

https://doi.org/10.36001/IJPHM.2023.v14i3.3123

1. INTRODUCTION

The electronic components and amount of data being produced from vehicles has increased exponentially. Advances in computing and storage capabilities, and connectivity, has created opportunities to leverage data, analyze any potential failures, unearth hidden patterns, and derive insights in an automated manner using ML which could help OEM take informed decisions.

The data values of vehicle sensors indicate the underlying behavior of the system. If the sensor values are as per the expected behavior of the system, then the data could be termed as 'Good' data and if the values indicate deterioration in the system, then it could be termed as 'Symptomatic' data. If this tagging information, whether data is good or symptomatic is present for each data point, then the data could be termed as 'labeled data' else the data is termed as 'unlabeled data'.

Depending upon availability of labels in data, the correct ML technique needs to be selected. ML is primarily of three types—Supervised, Unsupervised and Semi-supervised. Supervised ML requires the data to be labeled, while for unsupervised learning label information is not required. Semi supervised is a combination of the two where unsupervised modeling is utilized first to label the data followed by supervised model.

The telemetry data received from the vehicle may contain symptomatic data, but since it is not labelled in the context or would require huge amount of effort to label it, it is difficult to model it with supervised learning.

This paper proposes an Anomaly detection framework which could be adopted to capture any aberration in the vehicle component utilizing related data. This framework would not require symptomatic data; therefore, it can work with unlabeled data.

Anomaly in automotive data can be categorized into point and contextual anomaly. Point anomaly is when an individual data instance can be considered as anomalous with respect to

the rest of data. Contextual anomaly is when a data instance is anomalous in a specific context but not otherwise (Chandola, Banerjee & Kumar, 2009). The anomaly detection framework we are proposing works well on both point and contextual anomaly.

This framework proposes unsupervised one-class classifier algorithm considering the data is unlabeled (Amer, Goldstein & Abdennadher, 2013). To reduce the complexity and dimensions of the data, statistical techniques of Principal component analysis (PCA) is also utilized (Telgaonkar & Deshmukh, 2015).

The output of anomaly detection algorithm could be a label or a score. A label classifies data points directly into category of anomalous or non-anomalous whereas a score quantifies the extent of anomaly in the vehicle component with a numeric score which is compared to a threshold value. Values above threshold are marked as anomaly. Additionally multiple thresholds marking low, medium, or high could be defined to indicate severity of anomaly in the component. Further Vehicle Health Index could be created by combining severity of multiple components

The proposed framework utilizes score method to define the anomaly, as apart from flagging an anomaly it also provides the extent or severity of anomaly, and hence is superior to label method.

The anomaly detection output could be consumed using methods such as Interactive, Semi-Autonomous and Autonomous (Theissler, 2013). In interactive mode no algorithm is used, and user does a manual analysis using created charts and graphs. In Semi-Autonomous mode the algorithm provides an output which flags a data point as anomaly. This needs to be confirmed by a domain expert before triggering any counter measure. In autonomous mode the anomalies are flagged, and action are taken automatically without any intervention. Interactive mode is time consuming as lot of manual analysis needs to be done to analyze all data. In comparison, semi-autonomous mode filters the anomalous scenarios, and the domain expert needs to inspect only filtered scenarios, confirm the anomaly, and give feedback on model accuracy. Once the model is perfected to catch such anomalous scenario it could be moved to Autonomous mode. For this framework we are utilizing semi-autonomous mode of consumption which could be converted to autonomous after achieving expected accuracy.

Once an anomaly is confirmed, it could provide label for the data. Our framework then also proposes to create a PM model for specific component utilizing this data for better accuracy.

The rest of the paper is organized as follows. Section 2 highlights the Anomaly Detection Framework describing different stages of the framework in brief. Section 3 provides Automotive Industry Example covering the system detail and Case study upon which we have applied this framework. Section 4 is for Conclusion and further discussions.

2. ANOMALY DETECTION FRAMEWORK

A vehicle has many different components, and each component could have many different parameters associated with it which describe its behavioral pattern. For e.g., Engine and ABS could be two different components in a vehicle. Engine itself could have many parameters such as RPM, temperature, speed etc. whose individual range and association with each other would define the behavioral pattern of engine.

A component or system when operating normally without any fault, would exhibit certain behavioral pattern. The same component when about to fail, its conditions would start to deteriorate and likely to exhibit patterns which is significantly different from its normal operating pattern. An anomaly detection framework could help in identifying these deviations quickly and accurately and also help to narrow down the problems areas.

We are proposing a framework which creates an anomaly detection (AD) model on unlabeled data and further utilizes this label information to create a Predictive Maintenance (PM) model.

The framework consists of following stages—

1. Data Processing
2. Data Analysis
3. Regime Identification
4. Modeling
5. Model Validation and Deployment
6. Data Labeling

Data Processing

During data processing stage, a pipeline is established where the vehicle data is ingested into the cloud or any other infrastructure, extracted in usable form, transformed, and processed for further analysis.

The received data could be ingested in real time as streaming data or as batch in text or XML format or any other flat file format. It needs to be processed to get the parameter values. Also depending upon the frequency of data, transformation such as aggregation of data and extraction of important features needs to be done. Data aggregation is important because at a very granular level data points may have fluctuations leading to inconsistent prediction result. Hence The incoming data is to be aggregated at a frequency which is neither too big nor small to capture any regime changes effectively in data (generally at 30 sec- 1 min). (Regimes are discussed in more detail in further sections)

Data Analysis

During data analysis the data parameters can be viewed in the form of graphs and charts. Pre-defined dashboard templates could be created for data analysis which would provide more insights into the data and help in identifying important variables for further analysis. PCA is used to reduce the complexity and dimensions of the data which helps in better

analysis. It works by generating fewer, new dimensions by creating linear combinations of the original data such that the variation in data is preserved and the new dimensions are uncorrelated to each other. This reduces the total number of dimensions while minimizing information loss and helps in simplifying the data analysis process.

Regime Identification

Many times, the behavioral pattern of a vehicle component is very different for different operating conditions (Regimes). For e.g., a vehicle carrying high load may exhibit different levels of rpm, fuel trim, throttle position or injection pulse compared to when the load is minimum. It is important to identify different operating characteristics or regimes of the vehicle and create separate local anomaly detection models rather than one global model, for covering complete known operating range. This would ensure that contextual anomalies are caught by the model and also leads to better accuracy as regime specific customizations are considered during model building.

If an operating range is not defined during the design phase and hence is not covered as part of regime identification, it would need to be identified post deployment and need to be defined during next cycle of local model update/creation. We have discussed more about the possible approach to do this in Section 4- Conclusion

Finding the regimes by only looking at few variables in a chart may be difficult. This could be done by checking the graphs of top few Principal Components with each other. Different operating regimes are likely to be seen as separate groups in these charts. If these regimes are clearly distinguishable in the groups, then rules for distinguishing factor needs to be charted and each datapoint is then assigned to a regime by running through those rules. If the regimes are not clearly distinguishable then unsupervised clustering algorithms should be run on the top ranked PCs to find separate clusters and mark each of those as a regime. This regime identification process needs to be implemented as a component

During final solution deployment, this regime identification component needs to be deployed to classify the incoming aggregated data into one of the regimes and then do anomaly detection through that local regime model. This would ensure that even if the incoming data is continuously changing its regime, it gets addressed by correct model.

Modeling

Once the data parameters are analyzed and important ones are identified, anomaly detection algorithms such as Local Outlier Factor (LOF), Isolation Forest, one-class SVM etc. could be applied on the data.

The model is trained on the data from telemetry device. For training, we utilize data which is free from any known or new faults or deterioration and hence could be considered as good data. To extract this good data, data points containing any failures, DTC or symptoms needs to be filtered out to get rid of known problems. Removing unknown or new faults is difficult. Reasonable level of accuracy can be achieved by a combination of one or more of following activities—

- Filtering out data points containing any failures, DTC, or symptoms to get rid of known problems
- Comparing data from multiple similar vehicles to check for similar data patterns and also getting the vehicle data validated by an SME Ensuring absence of any deterioration or symptoms.
- Driving the vehicle in controlled conditions
- Checking with driver to ensure that no malfunction and deterioration were observed during the usage

Above measures would help in ensuring that data utilized for training is good data.

Similarly, we also have the test data which ideally should be a combination of Good and Symptomatic data (Anomalous). In absence of symptomatic data for testing the model, synthetic data generation techniques can be used to create anomalous data. A good model would be one which would be able to differentiate between Good and symptomatic data with high accuracy in test data.

The anomaly detection model is trained on train data, and it would provide a label or score against each data point. If it's a score, a threshold value needs to be decided such that, it is above the score of most of the data points in train. Utilizing the trained model, prediction is done on test data. The threshold value is now compared with score of test data and all the points above the threshold are marked as anomaly. Alternatively, if the severity of anomaly needs to be marked then multiple threshold values could be defined indicating level of severity, for e.g., low, medium, or high.

The final output from this stage would be prediction label on each data point in test data.

Model Validation and Deployment

It is important to note that not all anomalies could be attributed as failure, therefor any deviation found needs to be validated and confirmed by a Subject Matter Expert (SME) before being considered as a likely failure. Alternatively, if labels are available on the data, whether it is good or anomalous then accuracy of model output needs to be measured against the original labels. If there are multiple models being considered, then best model based upon a defined performance metrics needs to be chosen. This validation of model is semi-autonomous as it needs to be vetted.

Once the model has been validated on test data for accuracy, it could be deployed in production. This would move the model from semi-autonomous to autonomous mode. The deployment can be done either offboard in cloud environment or onboard on vehicle itself.

When the system or subsystem encounters any scenario which is aberration from its normal behavior, the deployed model would produce an alert and then appropriate action can be triggered.

Post deployment, there may still be rare instances of new vehicle operating regimes which are not defined during the design phase. Such instances would also be reported as anomalies by the system as it won't confirm to the established standards however these needs to be categorized as an operating regime. A likely approach to achieve this has been discussed in Section 4- Conclusion.

Data Labeling

The anomalous behavior could be a symptom of a known problem which is likely to occur, or it could be a new(undefined) failure type. This needs to be confirmed by an SME. Further, depending upon the criticality, this anomalous data information could be utilized for labeling the data so that a more accurate supervised predictive maintenance model could be created from it.

The entire framework is summarized in the chart below.

Data Processing
- Data Ingestion into platform - Applying Data transformation techniques
↓
Data Analysis
- Visual data exploration - Dimensionality reduction for large data
↓
Regime Identification
- Identifying local regimes for full operating range by applying domain knowledge and statistical methods
↓
Modeling
- Divide data into > Train - Good data > Test data - Symptomatic + Good data - Apply multiple algorithms and Create models for all local regimes
↓
Model Validation and Deployment
- Compare confusion matrix for different models - Deploy the best performing model - Identify "Not defined" regimes and provide feedback for model improvement
↓
Data Labeling
- Utilize the predicted anomalies to label the data - Supervised Predictive maintenance model could be created based on the criticality of anomaly

Figure 1. Flowchart explaining the framework

3. AUTOMOTIVE INDUSTRY EXAMPLE

3.1. System Details

The data on which the framework is applied, is from a fleet of electric off-road vehicle used for carrying load. The data is collected via a telematics device. The target component is hydraulic system, which is utilized to lift weight. For this it uses pressurized oil to generate movement. The system comprises of an oil pump which pumps the oil to cylinder. A control valve regulates the amount and pressure of oil needed to move the cylinder up and down. The oil circulates via return line and is stored in a reservoir tank from where it is filtered and fed back to the oil pump.

Please refer the Figure 2 below:

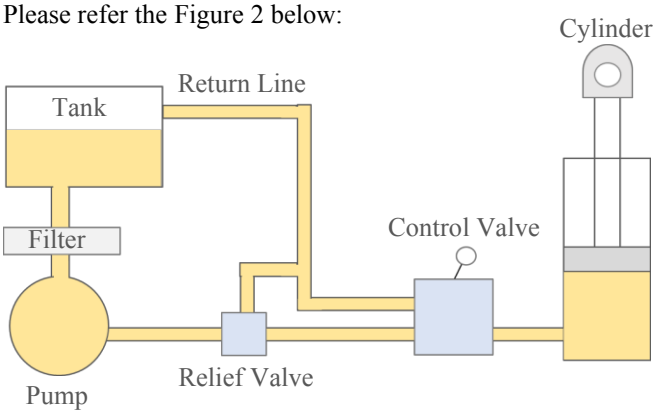

Figure 2. Vehicle Hydraulic System

3.2. Case Study and Results

Objective

In this case-study, we are trying to do anomaly detection in hydraulic system and further create a predictive maintenance model using generated labels. The fleet data utilized had more than 40 vehicles collected over a year and consisted of multiple session data. The sampling frequency of data was 45 parameters coming in per second. The data had fault information available in it. For the anomaly detection framework, this information was not utilized during modeling, however it is used for confirming the anomaly once flagged by the framework.

The framework required two sets of data for training and testing the model respectively. The training data only contained good data as described in Section-2 (Modeling). Since the labels for the data were not directly available, we filtered out the good data as one which did not have any DTC logged in the vehicle, to ensure removal of known problems. We also compared data from such vehicles which each other, which also showed similar patterns and hence minimized the chances of having known problem symptoms or new problems. This good data was divided into separate vehicles with 80:20 ratio. To prevent data leak between train and test, a vehicle's data was utilized either for train or test but not

both. The vehicles with 80 percent of the data were utilized for training. For test data we combined data consisting of DTCs and symptoms related to Hydraulic system and remaining 20 percent of good data. This ensured that test data had combination of Good and Symptomatic data.

Any cloud environment could be utilized for implementation of this framework. This case study was realized using Azure Data Factory pipeline. Following activities were employed for each of the framework stages—

Data Processing

The frequency of original data was per second because of which it was highly sensitive. To smoothen the data, it was aggregated on a 30 second tumbling window. The total sample count in the final dataset was close to 25K. Feature engineering was done on data to create features of Mean and Standard deviation for continuous variables. The data was marked for training and testing as explained above. This data was then stored for consumption for next stage.

Data Analysis

During data analysis graphs between individual variables were explored and variables related to Hydraulic system were identified. Since the number of variables were high, Principal Components (PC) were created for these variables and plotted for further exploration.

Regime identification

Next, different operating regimes were to be identified from the data. On plotting PC1 vs PC2 (Figure 3) we found that there were two main operating regimes. On further analysis of original data points, it is found that the differentiating parameter between both the regimes was the vehicle load. The left regime contained the datapoints when the vehicle was not lifting any load and right regime contains the datapoints when the vehicle was lifting load. This indicated that the characteristics of the vehicle became very different when operating under load and hence it is required to model each regime separately.

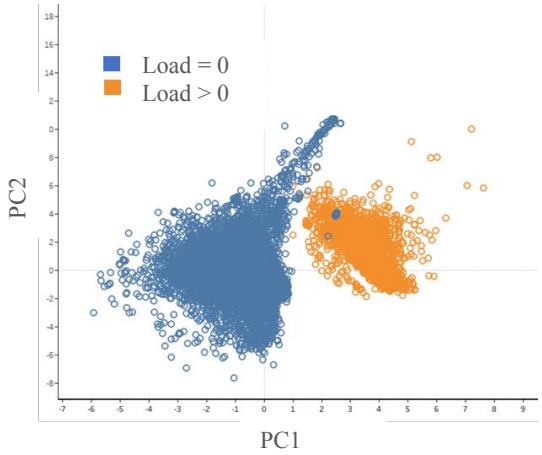

Figure 3. PC1 vs PC2 plot showing two different regimes

These regimes were then marked for each point in the data. The next stage of modeling was done for each regime separately.

Modelling

For modelling we labeled train and test data as described in "Section 3.2 Objective". For test data we have taken a combination of good vehicle data and data from vehicles having DTC and symptoms. The symptoms are likely to show up some time before the occurrence of the DTC. This duration could vary from one problem to other. In this case, the symptom data as confirmed by SME has been marked for up to two hours before the occurrence of the DTC.

We take the first regime where load is absent and train the model utilizing three different algorithms namely- one-class SVM, Isolation Forest, and Local Outlier Factor. The trained models were then used to predict on both train and test data. The prediction values were in the form of score for one-class SVM and directly a class label for others.

As mentioned in section 2, since the output is a score therefor model can be further tuned. Based on these scores, multiple thresholds were tried, and ideal value was finalized such that maximum training datapoints which are not isolated cluster in the graph, lie under the threshold value. Utilizing this threshold, prediction labels were marked on test data and model's effectiveness to identify outliers is visualized. As an e.g., four different threshold values and subsequent labels derived from one-class SVM model are plotted in Figure 4 below.

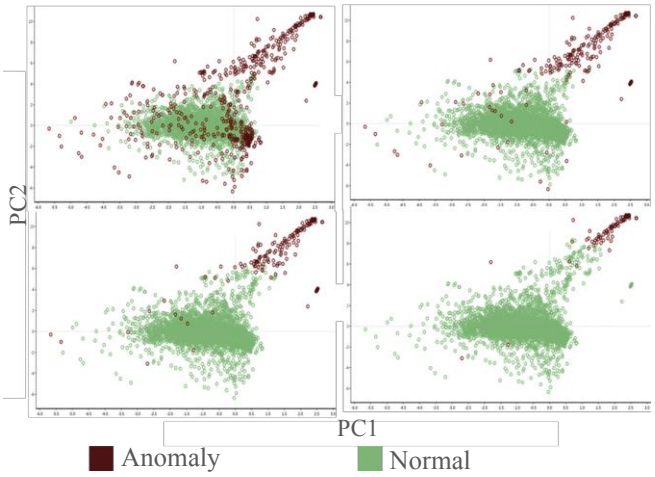

Figure 4. Labels with different threshold values 1.2 (top left), 1.8 (top right), 2.55 (bottom left), 4 (bottom right) respectively

The threshold value may be tweaked as per the required effectiveness on test data. This is done in iterative manner. The final output from this stage is a prediction label on each data point in test data set for each of the three algorithms.

Model validation and Deployment

To identify the best performing model among the three models, we compared their performance in identifying anomaly in the test data.

Provided below in Figure 5 is the confusion matrix for each of the algorithm on the test data.

One class SVM		
	Predicted Normal	Predicted Anomaly
Normal	86.12%	3.15%
Anomaly	0.73%	10.00%
Isolation Forest		
	Predicted Normal	Predicted Anomaly
Normal	83.51%	5.76%
Anomaly	1.27%	9.46%
Local Outlier Factor		
	Predicted Normal	Predicted Anomaly
Normal	83.98%	5.29%
Anomaly	1.16%	9.57%

Figure 5. Result comparison of different algorithms

Figure 6 provided below depicts KPI's of accuracy, false positive and false negative.

	One Class SVM	**Isolation Forest**	**Local Outlier Factor**
Accuracy	96.12%	92.97%	93.56%
False Positive	3.53%	6.45%	5.92%
False Negative	6.77%	11.84%	10.78%

Figure 6: KPI Comparison

As could be seen from above metrices and figures, we can conclude that the best performance was achieved by one-class SVM with least number of false positives and best accuracy.

We now mapped the DTCs in the PC1 vs PC2 (Figure 7) to check visually if these were captured by the SVM algorithm.

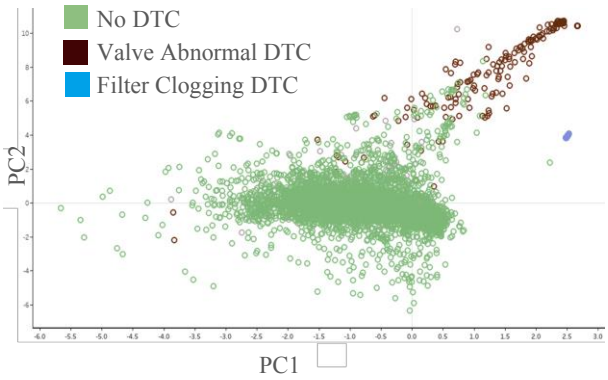

Figure 7. DTCs present in test data

We also created timeseries graph for each vehicle from test data against PC1. Anomalous points as derived from one-class SVM were found and were followed by DTC. One such vehicle data is shown in Figure 8. This provides a high certainty that the anomaly was indeed present as later it led to failure as can be seen on timeseries graph. This validates model's effectiveness in capturing anomaly.

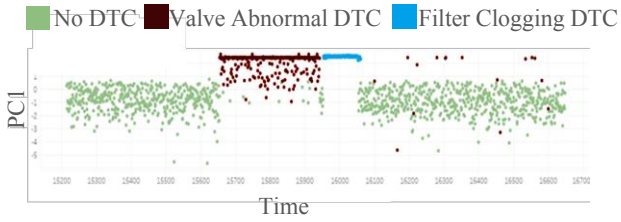

Figure 8. Timeseries plot of PC 1 displaying DTCs in the data

Comparing Figure 4, Figure 7 and Figure 8, we can derive that SVM captures the DTCs and also the symptomatic data before it as anomalies.

This model is now ready to be deployed onboard (edge) or offboard (web service), as per the requirement for anomaly detection.

Data labelling

As labels of failure are available from anomaly detection and confirmed by DTC presence, an early alert failure prediction (Predictive maintenance) model could be created for anomaly captured.

On analyzing the failure in detail, we found that anomalous points belonged to control valve abnormality as DTC related to control valve abnormality was also seen during this time (red color in Figure 8). Since the control valve is responsible for maintaining the pressure across the system, we checked the value of hydraulic pressure during this period and found it to be low (Figure 9), which indicates some abnormality with the Control valve. Few of the possibilities of abnormal control valve are Incorrect adjustments, Dirt and Particles holding the valve uneven or clogging of oil filter (which is a key maintenance item).

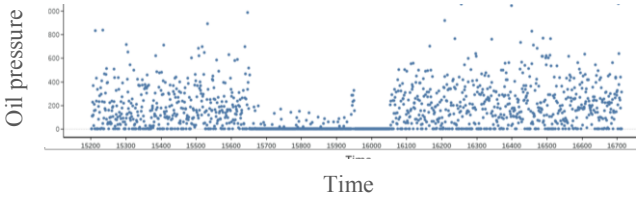

Figure 9. Oil Pressure on time series

Since, after Abnormal Control valve DTC was observed, oil filter clogging DTC were also observed in the data (blue color in Figure 8), this seems most plausible reason for control valve abnormality. We could safely conclude that starting of clogging of oil filter led to lower hydraulic pressure and control valve DTC and later substantial clogging resulted in oil filter clogging DTC. Hence, if low hydraulic pressure and control valve abnormality are caught early it can help predicting the oil filter clogging.

This predictive maintenance supervised model would be more focused and effective in capturing this specific anomaly than generic anomaly detection model. The decision to create a predictive maintenance model could be taken as per the criticality of the anomaly.

4. CONCLUSION

The framework helps in providing an automation solution to quickly analyze the field data and provide alerts for any aberration. It is useful in creating early alert model for any known problem or new anomaly in absence of labeled data. The infrastructure, pipeline creation, or models could be configured as per the specific requirements of the problem and is technology agnostic. The framework also proposes to convert a generic anomaly detection problem to specific predictive maintenance problem once the labels are captured in the data.

Additionally, this framework could be extended for improving future vehicle designs by incorporating any new fault type identified by the model.

Another aspect for which this framework could be utilized is for creation of a Vehicle Health Index (VHI) indicating the overall health of the vehicle. For this, model score from multiple components or system could be collected and provided a weightage based upon the criticality. These values could then be aggregated to provide a VHI score of the vehicle.

While the framework is likely to cover most defined scenarios, there could be rare instances when a specific operating condition is not defined during the design stage. These instances would be marked as an anomaly by the system as it won't belong to the established local model to which it has been assigned. One possible approach for identifying such scenarios could be that when these operating conditions occur, it is likely to occur as multiple instances and these instances should show as densely populated cluster of anomalies as it itself is an operating regime (As an operating regime defined by our framework should form a dense cluster). Therefor it is important that once a model has been deployed, its performance is monitored on a regular basis initially, especially when the points forming a dense cluster of anomalies are reported. During the performance monitoring, the SME would need to investigate such rare scenarios and confirm if it is new condition or actually an anomaly and if former, then going forward this data should be taken as feedback for improvement and considered for creating another local model for the new operating regime.

This approach needs to be further explored and researched for effectiveness.

5. LIST OF ABBREVIATIONS

Provided below in Table 1 are the list of abbreviations

AD	Anomaly Detection
DTC	Diagnostic Trouble Code
KPI	Key Performance Indicator
LOF	Local Outlier Factor
ML	Machine Learning
OEM	Original Equipment Manufacturer
PCA	Principal Component Analysis
PC	Principal Component
PM	Predictive Maintenance
SME	Subject Matter Expert
SVM	Support Vector Machine
VHI	Vehicle Health Index

Table 1. List of Abbreviations

REFERENCES

Amer, M., Goldstein, M. & Abdennadher, S. (2013). "Enhancing one-class support vector machines for unsupervised anomaly detection". *ACM SIGKDD Workshop on Outlier Detection and Description*, August 11-11, New York, USA, doi: 10.1145/2500853.2500857

Chandola, V., Banerjee, A. & Kumar, V. (2009). "Anomaly Detection: A Survey". *ACM Computing Surveys*, 41 (3), 58. doi: 10.1145/1541880.1541882

Telgaonkar A.H., Deshmukh S. (2015). "Dimensionality Reduction and Classification through PCA and LDA". *International Journal of Computer Applications*, 122 (17), 4-8

Theissler, Andreas (2013). *Detecting anomalies in multivariate time series from automotive systems.* Doctoral dissertation. Brunel University, London, UK. http://bura.brunel.ac.uk/handle/2438/7902