

**National College of Ireland**

**Project Submission Sheet**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** | Tushar Rajesh Gharpure | | |
| **Student ID:** | x23289902 | | |
| **Programme:** | MSc. Data Analytics | **Year:** | 2025 – 26 |
| **Module:** | Domain Application | | |
| **Lecturer:** | Vikas Sahni | | |
| **Submission Due Date:** | Apr 4th, 2025 | | |
| **Project Title:** | CA 1 | | |
| **Word Count:** | 3661 | | |

**I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.**

**ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.**

|  |  |
| --- | --- |
| **Signature:** | Tushar Rajesh Gharpure |
| **Date:** | 04 / 04 / 2025 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS:**

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

|  |  |
| --- | --- |
| **Office Use Only** | |
| Signature: |  |
| Date: |  |
| Penalty Applied (if applicable): |  |

# AI Acknowledgement Supplement

|  |  |  |
| --- | --- | --- |
| **Your Name/Student Number** | **Course** | **Date** |
| Tushar Rajesh Gharpure | MSc. Data Analytics | 04 / 04 / 2025 |

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click [here.](https://libguides.ncirl.ie/useofaiinteachingandlearning/studentguide)

# AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

|  |  |  |
| --- | --- | --- |
| **Tool Name** | **Brief Description** | **Link to tool** |
| **NA** | NA | NA |

# Description of AI Usage

This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

|  |  |
| --- | --- |
| **[Insert Tool Name]** | |
| [Insert Description of use] **NA** | |
| [Insert Sample prompt] **NA** | [Insert Sample response] NA |

# Evidence of AI Usage

This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

**Project Title: Optimizing Metro Train Maintenance With Predictive Analytics**

**Abstract**

This project is used to detect failures in the sensors of train Air Pollution Unit. Waiting to be failing of parts or sensor at crucial time or checking of everything at fixed schedule called as scheduled maintenance, we use sensor data installed in the metro train to detect the failures in future. We use this data to predict something might be wrong based on sensors data. There are various sensors and the compressed air is used for critical things like braking and suspension.

We made a deep learning model call sparse autoencoder that will learn the normal behavior of sensor and detect when something unusual happens. For example, if temperature sensor is not working properly and the data is unusual, the model can’t create data properly this will helps to spot the problems in the sensor.

This is a smart and efficient system that can be used to detect early signs of failure and pin the part which might be going to fail [1]. We also grouped similar kind of all problems that can cause trouble using clustering, we grouped similar failures together. This can be helpful because not all problems are same for example, some may dur to temperature sensor, or some may due to pressure sensors.

**Introduction**

In this day to day fast life we are totally depended on transportation basically on metro, trains, buses etc. To maintain this transport with high efficiency, safety and reliability this systems must be constantly monitored and must be watched closely. In this project we are gathering sensors information of Air Production Unit from Metro train dataset and used this information to predict failures. The Air Production Unit supplies constant pressured air which is used for braking and suspension and other activities [1].

A failure in the APU can lead to various critical problems like malfunction, delay, and safety issues. Basically we are using sensor data for predictive maintenance, which will allow us to carry out maintenance when it actually needs to.

Our model not only detects abnormal behavior but also provides insights into which sensors are failing, how the patterns change over time, and how similar anomalies can be grouped into failure categories. By integrating time-series encoding, visualization, clustering, and explainability into the SAE model, we have developed a solution that is accurate, interpretable, and suitable for real-time deployment in metro train environments [1].

**Background and Scope**

Now a days automobiles are playing a crucial role in day-to-day life as they are the only main source of transportation. For public transportation metro train system are necessary in modern and smart cities [2]. As a automobile there are lots of components that worked together and forms a machine. There are different components that functions or works together and one of the main crucial part in the metro is Air production Unit (APU), which controls and maintain supplies of compressed air to various components or sub system in the train. There are lots of measures that monitor the Air Production Unit in train such as pressure sensors, temperature sensors, flow rate and other metrics are measured through sensors. If any of the components or sensors fails it can cause maintenance, service, delays and can raise a safety issues also.

There are two types of maintenance reactive and predictive maintenance. Reactive maintenance work fixings things after breakdown and preventive maintenance are done based on schedule. These methods can bring more errors and are less effective can lead to unnecessary repairs and breakdowns. The best solution to get rid of this problem is predictive maintenance that uses sensor data and can predict when a failure might occur so that it can be prevent or fixed to avoid the failure and breakdowns.

**Goal**

The main goal of our project is to focus on building a better predictive system to detect the metro train APU failures, especially in the Air compressor systems and we are doing this by applying improved deep learning model known as Spare Autoencoder [2].  
 We aim to design.

1. A time based deep learning model to detect the pattern with respect to time.
2. To show the activity of sensors to detect unusual behaviors.
3. Grouping of similar failures together using clustering.
4. Charts and graphs that explain what is the purpose of model and what model is detecting

Why this technique we are using? Sparse Autoencoder is a good technique that is mainly used for leaning patterns from normal and abnormal behavior from the data. When anything unusual happens the model detect the difference and helps us to detect problems. It is a light weight and reliable technique and best fit for unlabeled data

Why this dataset? Metro train data have real time data of sensors from the air production unit of a train. The data considering of both digital and analog sensor values used to detect problems and can be used to testing models for prediction equipment failures. It is unlabeled data which is perfect fit for unsupervised learning method.

**Background and Literature Survey**

Several research are already done by using the same dataset using different techniques including sparse autoencoder.

Several studies have applied machine learning, deep learning for detection of failures with the same dataset which consists of real world sensor dataset of Air Production Unit.

One of the study I’m considering motivation for this project is *Predictive Maintenance Based on Anomaly Detection Using Deep Learning for Air Production Unit in the Railway Industry*, where they used sparse autoencoder and compared the prediction with variational autoencoders (VAE). The results is more accurate by using of Autoencoder model. However, their work was limited in terms of time-series awareness [1].

The most recent study, titled *Predictive Maintenance of a Metro’s Air Compressor*, compared the techniques such as Logistic Regression, Random Forest,XGBoost and CatBoost on the same dataset. Although the gradient boost is giving the best results but all they are complex and computationally heavy, making them less practical as the data will be realtime [2].

The paper where all the sensors information is introduced to help researchers to build failure detection model based on train metro dataset *The MetroPT dataset for predictive maintenance* includes all the sensor data from train and records of real faults. These information is very useful for deep learning methods that can find problems effectively to avoid failures [3].

Considering this previous researches done with respect to the same dataset, we selected spare autoencoder to use in for this project in effective way. In one key research from where we are taking motivation titled *Predictive Maintenance Based on Anomaly Detection Using Deep Learning for Air Production Unit in the Railway Industry* the author used a sparse autoencoder on the similar dataset, however their model only processed data from point to point (2D input), without considering values with respect to time. Also, they did not include sensor level analysis or group failures into categories [1].

Our project builds upon the previous sparse autoencoder approach but introduces several key enhancements. We transform the input structure from 2D to 3D using time-windowed sequences (30 timesteps), allowing the model to learn temporal trends and gradual sensor degradation. We implement a TimeDistributed sparse autoencoder with L1 regularization to enforce sparsity, improve reconstruction quality, and reduce overfitting. In addition, we analyze per-feature reconstruction errors, which helps identify which sensor is malfunctioning. Finally, we cluster detected anomalies using PCA and KMeans to classify different failure types and provide maintainers with more specific fault insights [2].

Through these improvements, our model not only detects anomalies with higher accuracy (achieving a low MSE of ~0.0014 and a strong clustering silhouette score of 0.63), but also enhances explainability and real-time applicability, making it a robust solution for predictive maintenance in metro train environments.

**Ethical Concerns**

During the development of this project, we came across some of the ethical aspects. First our model is dependent on anonymous data which is collected from MetroPT dataset and it does not have any personal or sensitive information about the individual under all GDPR regulations with ethical data usage standards. However, this system will be used to predict the failure and anomalies which is used for predictive maintenance of trains used for public transport, etc. This is a critical and reliable model and we categorize or grouped the failures which is a useful information for maintenance team in terms of predictive maintenance management which will saves lots of delay and cost. This will improve transparency and helps build trust in system. If our model is deployed in real-time usage. It should be tested to make sure it performs equally well across different environments and sensors types. Overall, our model is designed with ethical responsibility in mind, focusing on safety, transparency, privacy, and fairness.

**Implementation of the Technique (Models)**

In this project we implemented an unsupervised deep learning technique called sparse autoencoder to predict and detect the anomaly that may or might happen in train APU unit from the sensor data. In this model, the sparse autoencoder is designed to learn the normal behaviors of sensors without needing of labeled data, making it useful for real world scenarios where failures are not denoted or prenoted [2]. Our model learn how system behaves in normal condition or under healthy condition and marks any number of data points that performs unusual pattern. To catch the changes in sensor behavior we used a technique called windowing, where the data is broken into segments of 30 consecutive readings. This will allows model to observe the sensors value more accurately with respect to time, rather than treating each reading.

Our base model is structured in three dimensions, number of samples, time steps(window size) and sensor features. For the processing of data we deployed TimeDistributed dense layers in TenorFlow/Keras, which applies same dense layers for each set of window or each time step in the input window. The autoencoder includes encoder that will compress the input in to lower dimensions and a decoder that reconstruct original input from compressed form. L1 regularization is used to improve model focus and this will include model to focus on most important sensor signals and due to this we can avoid overfitting.

By using MSE(mean squared error) for training at loss function we measured how much different is the original data and reconstructed data. A high reconstruction error suggests that the model is unable to recreate that input properly—indicating a possible anomaly. This method more robust and it can be adjusted automatically as per the data.

In this project we not only detected the failures, we also tracked the per feature reconstruction errors to pinpoint which specific sensors are contributing the most to each failures or anomaly. Further more we categorized the by using Kmean clustering to group the failure into different failures types. And according to this we created various visualizations. This visualization will help to maintenance team to work on the specific part of sensors and carry out maintenance [2].

**KMeans Clustering (with PCA)**

In our project we had used this KMean with PCA to analyze and categorize the anomalies into groups in our model. After we specified the model in to time windows as anomalies we saw the sensor behavior are unusual. Here we wanted to understand whether these anomalies were similar to each other or represented different kinds of faults. Next what we have done we use PCA principal component Analysis to reduce the size of features. Instead of looking at 17+ values we shrunk the data into dimensions and kept only the important information. After that to groups all the sensors i.e. 17 sensors in our case we use KMeans Clustering to group and categorize the sensors. This made easy to visualize.

For example:

Cluster might contain small pressure issues, Cluster 1 might contain high motor values, Cluster 2 might contain oil temperature values.

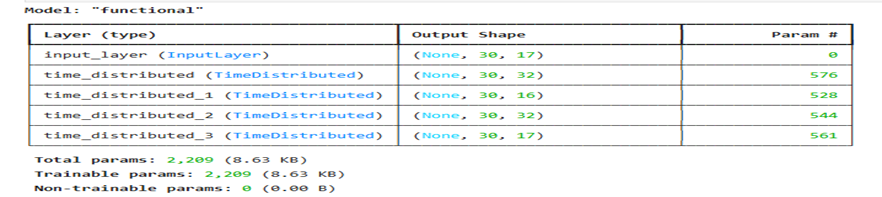


Fig. 01. Architect of 3D Model.

This is the architecture of our model sparse autoencoder, here instead using single point data we used multiple layers distributed with respect to time called TimeDistributed layers. This is helpful because sensor reading changes with respect to time and picking this patterns helps us to detect gradual failures like pressure drops or temperature rise that might not be obvious at a single point [1].

The input layer is distributed as (None, 30, 17). This states that the model expects input sequences 20 time steps (like 30 seconds) with 17 different sensor features at each step. The None means that in each batch it may have any number of sequences.

The next step we did is we have TimeDistributed layers these layer are fully connected at each time step. The first layer expands each timestep’s feature to 32 values using ReLU activation, which helps the model to learn some complex patterns. The layer compress to 16 where model learns a normal behavior of the sensors.

After this the decoder will reconstruct the data, it will first expands the from layer 16 back to 32 and finally brings at 17 features using sigmoid. Here the working will be like if the input is normal the reconstruction will be close but if its anomalous, the output will be different called reconstruction error and it will be high.

The total trainable parameters is 2,209 which is good for real-time data use. Here we used L1 regularization so that it will help to noise and focus on main signals, it do make model less overfit.

A graph with numbers and letters

AI-generated content may be incorrect.

Fig. 02 Reconstruction Error Distribution

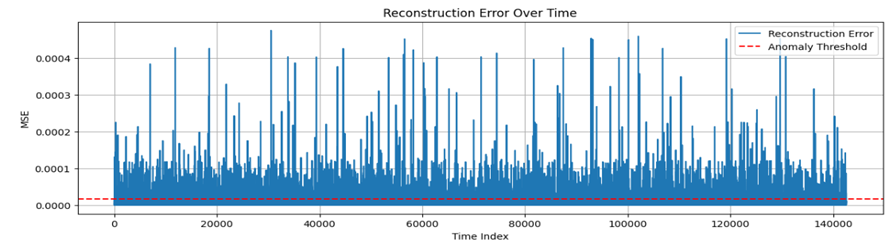


Fig. 03 Reconstruction Error over time.

The above image is the representation of Mean Squared Error Distribution. This visualization helps us to understand how model is performing for detecting the anomalies. On the x-axis we have MSE values which represents how much is the difference between the original values and the reconstructed sequences. The y-axis is based on showing the frequency of these error values across the entire dataset. We can observe that the most of the sequences have very low reconstruction errors, which confirms model has learn to accurately reconstruct normal patterns in the data. However there is a long tail on the right showing some evidence of sequences with unusual high reconstruction errors and this is called as anomalies.

Here we have set a threshold of 95% which means that any sequence with an error above 95% or threshold will be called as anomaly or failure. From our plot we can see that 7,128 out of 142,556 sequences were identified as anomalies. From this we came to conclusion that around 5% of data contain faulty behavior.

The second image is about reconstruction error over the time period. X-axis denotes the time index and y-axis denotes the Mean Squared Value (MSE) between the original sensor input and its reconstructed output for each time step. Here the blue line indicates reconstruction error at each point in time. The main purpose of plot is to give view of anomalies across the dataset. Here when autoencoder if trained on normal data it will learn the normal behavior and also reconstruct this type of input very accurately. If the reconstruction become poor the error will be spike, and here we can see it on vertical lines.

If the reconstruction error line falls above the threshold then it will comes under the category of potential anomaly.

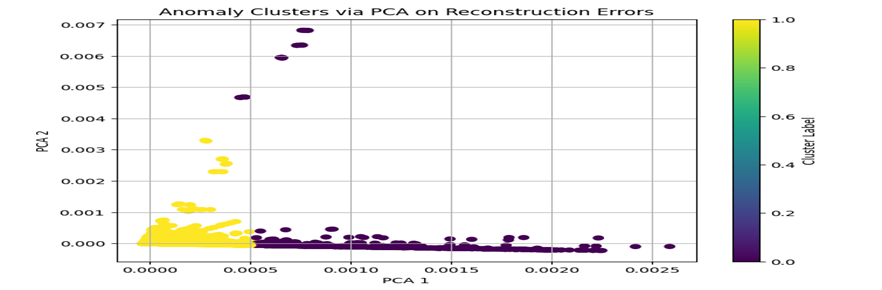


Fig. 04. Anomaly Clusters via PCA on reconstruction errors.

After identifying which data is anomalous we applied here principal component analysis and then Kmeans cluster to group the anomalies into distinct or to make the groups of failure categories.

On the x and y axis we have principal components (PCA1 and PCA2) which is the most important that capture the most of the reconstruction errors values and this are the values which is having more than 95% threshold. This scatter plot will helps us to visualize complex multi-sensor error data in a simple scatter plot, where each point represents anomalous sequence or we can say failures sequence.PCA reduces the data into two principal components—shown on the x-axis (PCA 1) and y-axis (PCA 2)—which capture the most significant variance in the reconstruction errors..

The anomalies are configured and colored according to the cluster label assigned by the KMean technique where each color is indicating different failure type or behavior category. The color bar is representing the cluster id’s showing anomalies are split into different id’s.It allowed us to group the anomalies into interpretable clusters, revealing underlying fault patterns. For example, one cluster might include sequences caused by gradual pressure loss, while another might relate to sudden spikes in motor current or irregular oil temperature readings. By doing visual inspection and then groupings and categorizing this, maintenance teams can understand not just that something went wrong but also what kind of issue it might be or which failure can happen in future.

The KMean will assign each cluster to a category, here each colored dot corresponds to different clusters of anomalies and given a unique label or category using kmean clustering technique. For example, one cluster might represent sudden pressure drops, while another could correspond to motor overheating or oil level issues. By categorizing anomalies this way, we move beyond just detecting something went wrong to understanding what kind of problem it might be.

In earlier research based on same dataset [2] they didn’t use or follow this kind of approach or technique to categorize failures or anomalies. We used this technique in our project to enable fault-type separation, which was missing in earlier research using similar datasets. Previous models would simply flag a data point as anomalous, but provided no insight into whether multiple failure modes existed or how they could be prioritized. With clustering, we can not only confirm the presence of anomalies but also organize them into meaningful categories for easier investigation, root-cause analysis, and targeted maintenance [2]. This will be helping maintenance team to find out the specific failure from the specific sensor.

A graph with a red line

AI-generated content may be incorrect.

Fig. 05. Zoomed Anomaly Detection.

The visualization is a detailed view of how our trained sparse autoencoder model performs anomaly detection over a small portion of data (300 points). This chart plots the reconstruction error (shown by the blue line) for each time step. Reconstruction error is calculated as the mean squared difference between the original input data and the output predicted by the model. In other words, it shows how well the model is able to recreate normal system behavior from its learned patterns.

This plot will allows us for fine-grained validation of the model’s sensitivity and specificity. Rather than looking at broad patterns across thousands of data points, we can clearly observe how individual abnormal readings are flagged. Secondly, it helps identify whether the detected anomalies are random spikes or occur in short sequences, which is crucial for understanding the nature of the failure.

A graph of a bar chart

AI-generated content may be incorrect.

Fig. 06. Boxplot of all Sensors values.

This plot is showing all the values for both digital and analog sensors such as TP2, TP2, H1, Oil\_temperature, Flowmeter, Motor\_current, etc. This plot shows us how each sensors behaves. Sensors like flowmeter, motor current and oil temperature sensors are showing higher values and wide variations that means they change a lot and carry important information and they may rise to key contributors for detecting faults. Other sensors like COMP, DV\_electric, LPS show very little change and mostly flat. Some of the sensors have tight ranges and maybe very stable example reservoirs and Dy\_pressure. Few sensors are showing outliers that means we already have unusual behavior in the raw data.

The purpose of using this boxplot in our predictive maintenance process was to identify potential data quality issues, such as outliers, and to guide feature selection for modeling. It also helped us identify sensors that behaves abnormal or values are less or more as compare to original data, which are more likely to be informative for the anomaly detection task. Moreover, the presence of high outliers might indicate either sensor faults or early signs of system anomalies or failure detection, further justifying the use of anomaly detection models like Sparse Autoencoders.

**The Quantitative Findings**

By using this dataset our model is performing very well for detecting of failures. We detect the unusual behavior of of sensors of train metro dataset. Our model is able to recreate normal sensor readings with low mean squared error values of around 0.0014 and a test RMSE value of about 0.00226. We maintained a threshold of 95% of reconstruction errors where if the values are beyond this threshold it will be falling under the failures or anomalies. Out of 142,000 data sequences our model detected 7,128 anomalies. The KMean clustering approach grouped anomalies effectively achieving a good silhouette score of 0.63.

The anomalies causing sensors are oil temperature, flowmeter, and motor current. More importantly, our model is not underfit nor overfit. Through various visualizations, we could clearly see when and where sensor errors occurred. This will help maintenance team to work on the sensor problems that may cause failure after some time.

**The Business Value of the Findings**

This project is offering strong practical benefits for maintenance team to reduce the further failures in trains. It will help to detect early issues and allows to repairs to be done before any major problem or breakdown occurs. This means fewer service delays and smoother train operations. By using the failure sensor data of which specific part or sensor is likely to fail we can schedule the maintenance more efficiently and this can avoid unnecessary checks [2].

The model also gives clear and desirable results making it easier for cost saving purpose by reducing downtime and focusing on major repairs to be done. This model is very light weight and fast and it is well suited for real-time data for actual in train metro system to avoid failures in future [1].

**Business Domain**

This project falls under the Transportation and Infrastructure sector or Automobile sector. So specifically focusing on Metro Rail Systems and Predictive Maintenance [1] we can detect the failures for maintenance to save delays and maintenance cost. While we focused on one specific metro system’s air compressor unit, the same methods and insights can be applied to other systems such as, locals, subway trains, light public transports, high speed train, electric trains and buses and machinery where compressed air is used.

This model supports smarter transportation, improves public service, and supports sustainability by extending the life of equipment and reducing unnecessary repairs.

**Reference**

[1] Najjar, Ayat, et al. *Predictive Maintenance of Urban Metro Vehicles: Classification of Air Production Unit Failures Using Machine Learning*. 2023, papers.ssrn.com/sol3/papers.cfm?abstract\_id=4403258, <https://doi.org/10.2139/ssrn.4403258>. Accessed 1 Apr. 2025.

[2] Davari, Narjes, et al. “Predictive Maintenance Based on Anomaly Detection Using Deep Learning for Air Production Unit in the Railway Industry.” *IEEE Xplore*, 1 Oct. 2021, ieeexplore.ieee.org/abstract/document/9564181/. Accessed 4 Apr. 2023.

[3] Barpute, Jyotsna Vilas, et al. “Predictive Maintenance of a Metro’s Air Compressor.” *2024 5th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 7 Aug. 2024, pp. 247–252, ieeexplore.ieee.org/document/10689744, <https://doi.org/10.1109/icesc60852.2024.10689744>. Accessed 1 Apr. 2025.

[4] Veloso, Bruno, et al. “The MetroPT Dataset for Predictive Maintenance.” *Scientific Data*, vol. 9, no. 1, 13 Dec. 2022, <https://doi.org/10.1038/s41597-022-01877-3>.