2025
DEVICE FAILURE
MONITORING DASHBOARD

BUSINESS INTELLIGENCE PROJECT

DATE :
APRIL 2025

PROPOSED BY :
ALI HOSSEINMARDI

LECTURER :
SAFA OLIA

The Hague University of Applied Sciences

Contents

Abstract

This report presents a Business Intelligence (BI) solution for monitoring device failure patterns using real-world sensor data. The project leverages data visualization and analytical tools within Power BI to identify critical insights that support predictive maintenance strategies. Through the use of KPIs, interactive filters, and advanced visuals—such as scatter plots, matrix views, and time-based trend charts—the dashboard enables clear understanding of failure distribution across devices, weekdays, and sensor values. Key findings highlight that Sensor A and Sensor F show a strong relationship with failures, while failures peak on specific weekdays such as Monday. The report also discusses ethical considerations related to data usage and proposes actionable recommendations to optimize operational efficiency and reduce downtime. This project demonstrates how BI can transform raw data into actionable insights for business decision-making.

1. Business Intelligence Scenario

1.1 Introduction

In today's factories and tech offices, machines and equipment are fitted with sensors to keep an eye on how they're running and spot problems. One important use of this data is predictive maintenance, which means using the data to predict when devices might fail so that organizations can reduce repair costs and minimize downtime. This project aims to study data related to device sensor readings and failure records. The goal is to build an interactive dashboard that helps stakeholders understand which devices are failing, when they are failing, and what patterns may contribute to failures. To do this, we did the following:

- Data Cleaning in Python: We used Python to explore the raw data, handle missing values, check for data types, and prepare the dataset for analysis. This step makes sure the data is clean and consistent before importing it into Power BI.
- Additional Cleaning in Power BI: While Python helped us with the initial cleaning, we also used Power BI to make further changes to the data. This included calculating important measurements like the failure rate and making sure the data model was suitable for visuals.
- Visualization in Power BI: We built a comprehensive Power BI dashboard to visualize device health and performance metrics. This includes trend charts, bar graphs, pie charts, and scatter plots to help users interactively explore failure patterns across time, devices, and sensor values.

2.Data Acquisition

2.1 Data Collection

The data used in this project was found on Kaggle.com, a website where people share data and challenges related to machine learning. The dataset has information about the readings from industrial equipment. The readings are recorded every day. The dataset includes information such as:

- Date: Information about when the observations were recorded.
- Device ID: Unique identifier for each machine or device.
- Failure Flag: A binary indicator (0 or 1) representing if a failure occurred.
- Failure Status: Text label that shows status of device. ("Healthy" or "Failure")
- Sensor A to Sensor I: Numeric values from different sensors monitoring internal parameters.

Before creating any charts or reports, it is essential to make sure the dataset is clean, complete, and correctly formatted. Raw data often has missing values, inconsistent data types, or irrelevant records. These can affect the accuracy of the analysis. We started by using Python to fix any technical issues. Then, we used Power BI to change the data, give the columns new names, and get the data ready to make a dashboard. Each row in the dataset shows how a device performed on a specific day.
The sensor columns contain numbers recorded by physical sensors inside each device. We don't have their official descriptions, but they likely measure things like:

- Vibration (e.g., if parts shake more than normal)
- Temperature (e.g., overheating)
- Pressure
- Humidity
- Current or voltage
- Mechanical strain

Each sensor can help detect small changes in the machine's behaviour. If the vibration level (Sensor A) or internal pressure (Sensor F) gets too high, that could be a warning sign that a device is about to fail.
By checking the readings over time, we can spot early signs of problems.

3. Data Transformation

Step 1: Cleaning the Data in Python

Before I used Power BI, we opened the raw dataset in Python using a tool called Jupyter Notebook. We used Python because it's great for checking and fixing large datasets quickly.
Tasks I did in Python:

- I checked for and removed any missing or incorrect values.
- Converted the date column into the proper datetime format for sorting and filtering.
- I sorted the records by date to prepare for time-based analysis.
- Finally, I exported the cleaned dataset as a CSV file for importing into Power BI.

Step 2: More Cleaning in Power BI

After fixing the basic issues in Python, we brought the data into Power BI to make more changes. Power BI has a tool called Power Query Editor, which lets us make final improvements before building charts:

1) Changing columns name to be more understandable. (e.g., metric1 → Sensor A, device → Device ID)
2) Changing data types, made sure "Date" is treated as a real date, not just text, "Failure" as whole number.
3) Removed empty rows or rows with all zero values that are not useful for analysis.
4) I also created new columns:
    - Day of Week: Shows which day the data was recorded (Monday, Tuesday, etc.)
    - Failure Status: A label to replace 0/1 with "Healthy" or "Failure"
5) I checked the sensors: Sensor A and Sensor F had the most useful data. Some other sensors (e.g., Sensor B, C, D) were mostly 0s.

4.Data Visualization

4.1 Dashboard Design and Key Performance Indicators (KPIs)

I designed a clear and interactive dashboard in Microsoft Power BI to help users easily understand and explore the device performance data. The dashboard provides real-time insights into which devices are most likely to fail, when failures occur, and how different sensor readings relate to machine health.

The dashboard is titled:"Device Failure Monitoring Dashboard". This name communicates the purpose of the dashboard to monitor device health and detect failure risks using predictive maintenance techniques.

We divided the dashboard into three sections:

1. **KPI Cards** (Top Section) These cards show the most important summary numbers:

- **Total Devices**: How many unique machines are being monitored.
- **Total Failures**: How many times a failure has been recorded.
- **Total Observations**: The total number of records in the dataset.

- **Failure Rate (%)**: What percentage of the records resulted in a failure.

These Kip's give users a quick overview of the overall machine performance.

2. **Main Visuals** (Middle Section)

This section contains different types of charts that help identify trends and compare devices:

Visual	**Purpose**
Bar Chart: Top 10 Failing Devices	Shows which machines failed most often.
Line Chart: Failures Over Time	Displays how failures changed from month to month.
Pie Chart: Failures by Day of Week	Helps identify if failures happen more often on specific days.
Column Chart: Failure Status by Day of Week	Compares healthy vs failed device counts by weekday.
Scatter Plot: Sensor A vs Sensor F	Helps explore if there's a relationship between two sensor readings. Larger values may suggest performance issues.

Table 1. Overview of Dashboard Visualizations and Their Analytical Purposes.

3. **Interactive Filters** (Slicers)To make the dashboard more flexible, I added slicers so users can filter the visuals based on:

- **Device ID**
- **Day of the Week**
- **Failure Status (Healthy or Failure)**

These filters allow users to explore specific machines, days, or failure conditions.

How KPIs Were Calculated (DAX Measures):

Total Failures = CALCULATE(COUNTROWS('Table'), 'Table'[Failure Flag] = 1)

Total Observations = COUNTROWS('Table')

Failure Rate (%) = DIVIDE ([Total Failures], [Total Observations]) * 100

Total Devices = DISTINCTCOUNT ('Table'[Device ID])

This dashboard is effective for some reasons:

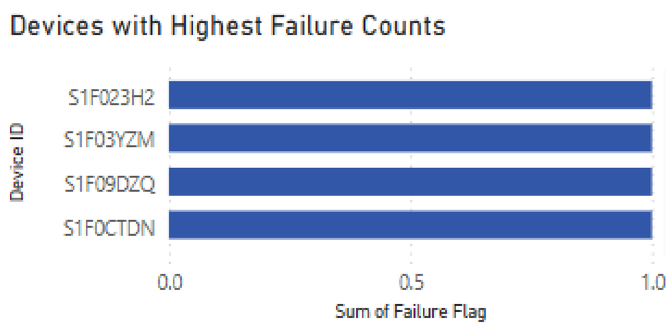
- It helps users quickly spot patterns in failure behaviour.
- It makes sensor data easier to understand, especially when comparing failed and healthy devices.
- It supports predictive maintenance planning, allowing maintenance teams to focus on high-risk machines.
- Even people with no technical background can use it thanks to its visual, intuitive layout.

5. Data Analysis

After getting the data ready and cleaning it, and designing the Power BI dashboard, I used different types of visuals to explore trends, find problem areas, and learn about device performance. Each chart or visual element was added for a specific reason to answer questions about failures, time patterns, and sensor behaviours.
Below is an explanation of the main visuals included in the dashboard and what we learned from each.

5.1 Top 10 Failing Devices - Bar Chart

This bar chart shows the devices that had failure events in the dataset. Each bar represents a different device ID. The value on the X-axis shows the total number of failures for that device. This is based on the Failure Flag (1 = failure, 0 = healthy).
Since failures are rare, most devices in this chart only failed once. But this chart is still useful because it shows which machines had any problems. This makes it easier to decide which devices to check or maintain, even if they only failed once. This chart allows you to quickly see how often failures happen on different devices and supports planning for maintenance based on risk.



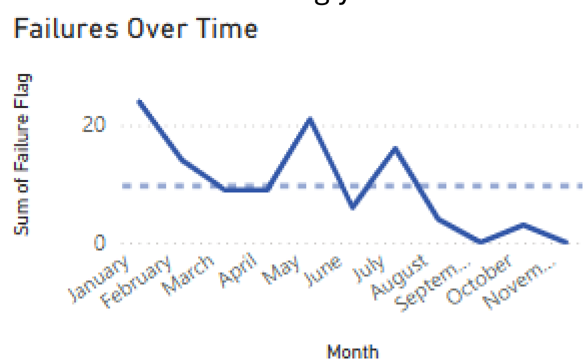
5.2 Failures Over Time - Line Chart

This line chart shows how device failures were distributed across the calendar year. The X-axis displays each month (from January to December), and the Y-axis represents the total number of failures recorded in that month based on the sum of the Failure Flag.
The chart reveals a clear variation in failure activity over time:

- **January** shows the highest failure rate, with over 20 failures.
- **May** also had a significant number of failures (around 20).
- In contrast, months like **September** and **November** had **no recorded failures at all.**

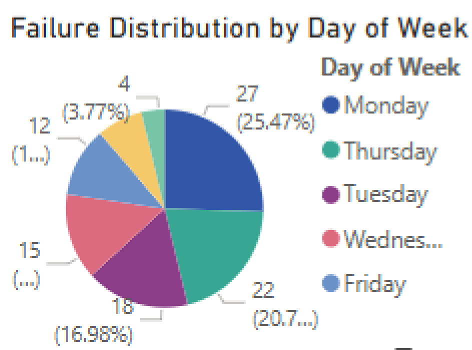
This type of trend analysis is valuable for identifying **seasonal patterns or operational cycles** that may influence equipment stress or breakdown risk. For

instance, higher failure rates at the beginning of the year could relate to post-holiday startup phases, increased production, or reduced maintenance over year-end closures.
The visual helps maintenance planners and managers **predict high-risk periods** and schedule proactive maintenance accordingly.



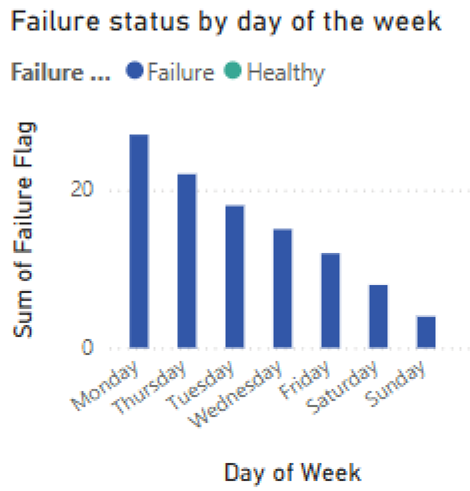
5.3 Day of the Week Failures – Pie chart

This chart shows the percentage of outages that occurred on each day of the week (Monday, Tuesday, etc.).
Most outages occurred on Mondays and Thursdays. This may reflect work schedules or busy times of the week when equipment is used more. It might be useful to schedule inspections on these days to catch early problems.



5.4 Failure status by day of the week – Column Chart

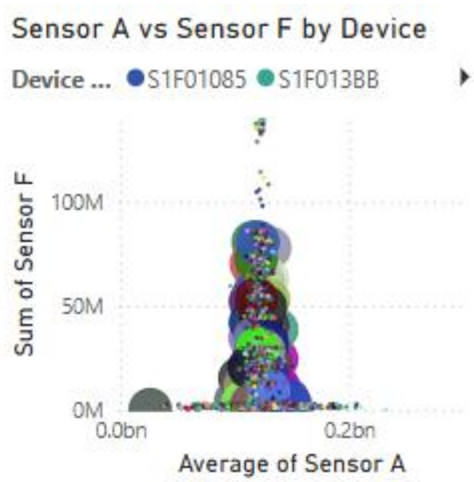
This grouped column chart compares the number of healthy and failed devices for each day of the week.
Insight: It confirms the finding from the pie chart and provides a side-by-side view. Mondays and Thursdays had higher numbers of failures. Other days had mostly healthy values. This adds more detail to the weekday patterns.



5.5 Sensor A vs Sensor F – Scatter Plot

Each point in the chart represents a device's sensor readings. The X-axis shows average values for Sensor A, and the Y-axis shows total values for Sensor F. The points are coloured based on device ID, and we also tested using bubble size to reflect the number of failures.

This chart helps explore the relationship between two key sensors. If a device has high values on both sensors, it may be under more stress or at risk of failure. Even though most devices didn't fail, the scatter plot allows us to detect unusual combinations of sensor behaviour that may lead to breakdowns.



5.6 KPI Cards – Summary Tiles

At the top of the dashboard, we included four KPI cards to summarize the most important metrics about the dataset:

- **Total Devices**: 1,169

- **Total Observations**: 124k
- **Total Failures**: 106
- **Failure Rate (%)**: 8.51%

These KPIs give users a quick overview of the system's health. In this dataset, although failures are relatively rare compared to the number of total observations, an 8.51% failure rate still suggests that some devices may need closer attention or preventive maintenance.

These indicators provide critical context for the rest of the dashboard visuals, allowing users to immediately grasp the **scale of operations** and the **frequency of failure** across the system.

5.7 Sensor and Failure Overview

Table This visual is a table that presents a combined view of key information for each recorded date. The table includes the following columns:

- **Date**: The day on which the data was recorded.
- **Device ID**: The unique identifier of each device being monitored.
- **Sum of Sensors**: The total value obtained by adding readings from all sensor columns (Sensor A to Sensor I).
- **Total Failure**: Indicates whether a failure occurred on that date (based on the Failure Flag value).

This table allows users to **quickly scan across devices and days**, helping them identify patterns in sensor behaviour and failures. By including the **sum of sensor values**, we can also get an idea of the device's overall activity level or internal stress on that day.

The table also supports **conditional formatting**, such as using colour to highlight columns when sensor totals exceed certain thresholds. This helps maintenance personnel easily spot abnormal conditions without reading every row manually.

While charts are useful for high-level trends, this detailed table acts as a **reference layer** — it gives context to other visuals and helps users verify and explore the data behind the summaries.

Date	Device ID	Total Failures	Sum of Sensor A	Sum of Sensor B
Thursday, July 16, 2015	W1F14XGD	0	100788520	64968
Sunday, October 25, 2015	S1F0GCED	0	18115368	64792
Monday, October 26, 2015	S1F0GCED	0	15068192	64792
Tuesday, October 27, 2015	S1F0GCED	0	87464512	64792
Thursday, October 29, 2015	S1F0GCED	0	29878480	64792
Friday, October 30, 2015	S1F0GCED	0	136903912	64792
Saturday, October 31, 2015	S1F0GCED	0	183224608	64792
Monday, November 02, 2015	S1F0GCED	0	94904784	64792
Monday, January 05, 2015	W1F0PNA5	0	38285364	64784
Tuesday, January 06, 2015	W1F0PNA5	0	65311277	64784
Total		**106**	**15236584525025**	**19855885**

5.8 Device Failures by Day of Week – Matrix

This matrix visual provides a crosstab view of failure activity by showing how many failures each device experienced on each day of the week. In this matrix:

- **Rows** represent each individual **Device ID**.
- **Columns** represent the **Day of the Week** (e.g., Monday, Tuesday...).
- **Values** show the **count of failure status**, based on how many times each device failed on each specific day.

This format makes it easy to compare patterns across devices and weekdays in a compact grid. For example, users can quickly identify if a certain device tends to fail more often on Friday, or if another device shows failures spread evenly across the week.

The matrix is especially helpful when scanning for **recurring issues** tied to specific devices and operational days. It complements the bar and pie charts by offering a more granular breakdown of failure frequency, and supports faster, data-driven decisions about **when and where to apply targeted maintenance**.

Device ID	Friday	Monday	Saturday
S1F01085	1	1	1
S1F013BB	1	1	1
S1F0166B	1	1	1
S1F01E6Y	7	7	7
S1F01JE0	1	1	1
S1F01R2B	32	32	32
S1F01TD5	1	1	1
Total	**18041**	**17886**	**17897**

5.8 Summary of Insights

We used different visual elements to look at the dataset in different ways. This let us:

- Find devices that are at high risk of failure

- Find time periods when more devices fail

- Compare how sensors behave when they are working or broken

- Find out which days might be risky

This kind of visual analysis helps us plan for maintenance, use our resources well, and make better decisions.

5.8 Conclusion

This project demonstrates how data science and business intelligence tools can be used together to improve maintenance operations in industrial environments. Using Python, we cleaned and prepared sensor data collected from various devices and then visualized that data using Power BI to create an interactive dashboard.

The dashboard helped us analyse failure trends, identify risky devices, and detect unusual patterns in sensor behaviour. While the overall system appeared stable, the

few failure cases we observed provided valuable insights into when and why devices tend to break down.
By translating raw sensor readings into meaningful visuals, we made complex data understandable for managers, maintenance teams, and other stakeholders who may not have a technical background.

5.9 Recommendations

Based on our findings and dashboard analysis, we suggest the following recommendations:

- Focus preventive maintenance on the top failing devices.
- Monitor Mondays and Thursdays more closely, as failure rates were higher on these days.
- Pay special attention to Sensor A and Sensor F readings, especially when both are unusually high.
- Continue improving the dashboard by adding more sensor comparisons or failure prediction models in the future.
- Extend the time range of the dataset for more comprehensive trend analysis.
- Train non-technical staff to use slicers and filters in Power BI for custom insights.

With these recommendations, businesses can reduce equipment downtime, plan better maintenance schedules, and avoid unexpected failures. The tools and methods demonstrated in this project can be easily scaled or customized for other types of machines, industries, or operational goals.

6.Predictive & Prescriptive Analytics

Although the main focus of this project is descriptive Business Intelligence (BI), it can be further improved by incorporating **predictive and prescriptive analytics**. These methods help organizations not only understand what has happened, but also what is likely to happen in the future — and what actions should be taken as a result.

Predictive Analytics – What Could Happen?

By analyzing historical sensor data and failure patterns, machine learning models (e.g., decision trees, logistic regression, or time-series forecasting) could be trained to predict:

- Which devices are most likely to fail in the near future
- The estimated time remaining before a failure occurs
- Which sensor combinations (e.g., high Sensor A and F values) are early warning signs

This would allow for smarter scheduling of inspections and part replacements before failures occur, helping avoid downtime.

Prescriptive Analytics – What Should We Do?

Prescriptive analytics can recommend optimal maintenance plans by combining predictive models with business rules and operational constraints. For example:

- Schedule maintenance **before** Monday or Thursday if a device shows rising sensor levels
- Allocate more resources to devices with higher predicted risk
- Adjust operating thresholds dynamically based on predictive alerts

These approaches help organizations align analytics with **short-term actions (weekly maintenance)**, **mid-term planning (seasonal performance)**, and **long-term goals (cost savings, reliability improvements)**.

7. Social, Ethical, and Environmental Considerations

Using Business Intelligence and sensor data analysis must be approached with care, especially when the outcomes can affect people, systems, and the environment.

Social & Ethical Considerations:

- **Transparency**: Operators and engineers must understand how insights are produced. BI tools should not be black boxes.
- **Fair Use of Data**: Data should only be used for the purpose it was collected. Avoid over-monitoring or misinterpreting behavior based on incomplete data.
- **Job Impacts**: Predictive maintenance may reduce emergency repair jobs, so workforce roles must evolve with proper training.

Environmental Considerations:

- **Resource Optimization**: Reducing unplanned failures minimizes wasted materials and energy from emergency repairs.
- **Sustainable Operations**: Avoiding equipment overuse or failure helps machines run efficiently, reducing carbon footprint.
- **Waste Reduction**: Predictive maintenance extends machine life and reduces unnecessary part replacements.

By considering these aspects, the BI project not only supports operational efficiency but also contributes to **responsible, human-centered, and sustainable technology use**.

8.References

1. Chien, C.-F., Ding, K., & Kuo, C.-Y. (2019). Data science and AI for smart manufacturing and industrial engineering. *Computers & Industrial Engineering, 128*, 905–915. https://doi.org/10.1016/j.cie.2018.12.004
2. Kaggle. (n.d.). *Predictive maintenance dataset*. Kaggle. Retrieved April 14, 2025, from https://www.kaggle.com/datasets/shivamb/machine-predictive-maintenance
3. Microsoft. (n.d.-a). *Power BI documentation*. Microsoft Learn. Retrieved April 14, 2025, from https://learn.microsoft.com/en-us/power-bi/
4. Microsoft. (n.d.-b). *DAX in Power BI*. Microsoft Learn. Retrieved April 14, 2025, from https://learn.microsoft.com/en-us/dax/
5. Project Jupyter. (2015). *Jupyter Notebook* [Computer software]. https://jupyter.org/
6. Python Software Foundation. (2024). *Python (Version 3.11)* [Computer software]. https://www.python.org/
7. Wang, K., Zhang, C., & Yu, L. (2020). Predictive maintenance using machine learning: A review. *IEEE Access, 8*, 181749–181768. https://doi.org/10.1109/ACCESS.2020.3025754

9. Appendixes

9.1 Python codes from Jupyter:

```python
# -----------------------------------------
# Import required libraries
# -----------------------------------------
import os
import pandas as pd

# Set the working directory
os.chdir("E:/BI/week0")
# -----------------------------------------
# Load the original raw dataset
# -----------------------------------------
df = pd.read_csv("predictive_maintenance_dataset.csv")

# -----------------------------------------
# Clean the 'date' column
# - Convert to datetime
# - Coerce errors into NaT (Not a Time)
# - Drop rows with invalid dates
# -----------------------------------------
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df = df.dropna(subset=['date'])
df = df.sort_values(by='date')

# -----------------------------------------
# Preview data
# -----------------------------------------
print(df.info())
print(df.head())

# -----------------------------------------
# Save cleaned dataset for Power BI
# -----------------------------------------
df.to_csv("predictive_maintenance_cleaned1.csv", index=False)
```