THE UNIVERSITY of ADELAIDE

Software Engineering & Project (COMP SCI 7015)

Rail Break Prediction ML

Initial Report

By **Group RAIL PG-2**

Tao Xu a1937511

Sheng Wang a1903948

Jinchao Yuan a1936476--

Zi Lun Ma a1915860

Di Zhu a1919727

Xin Wei a1912958

Yifan Gu a1909803

Tianhua Zhang a1915934

Zihan Luo a1916700

Supervisor : Murtaza Bootwala

1. Project Vision

The project aims to build a data processing pipeline and a machine learning model to detect possible rail breakages in the Australian railway network. All the extensive sensor data is provided by the Australian Rail Track Corporation (ARTC), and InsightFactory platform will be applied for accomplishing data initiatives. After data is processed, multiple machine learning methods will be used to find patterns in it and make predictions. The main goal is to create models that are accurate, stable, and fast. The project will use SQL and Python to process data and train models. The final outcome will provide useful insights to the ARTC to improve safety, reduce delays, and keep freight operations running smoothly.

2. Customer Q&A

2.1. Can we assume all code execution will run in the cloud instead of in our local environments?

Yes, all code execution you perform on the platform will run on their computer cluster, not on your local laptop. Therefore, having a high-performance laptop with a good CPU will not make any difference.

2.2. For the project management of this project, can we use JIRA?

No, the team will use the project board in GitHub, which functions similarly to JIRA. All task management and related activities will be handled in GitHub to keep everything within a single platform.

2.3. Regarding the project architecture, my understanding is that there are three layers: the data ingestion layer, the data processing layer, and the machine learning layer. Is that correct?

Yes, that is correct.

2.4. So we may have multiple machine learning models for the machine learning layer, right?

Yes, especially at the start. Since the project involves research, it's not recommended to pick one model at the beginning and use it for the entire project. In the first two sprints,

the focus will be on exploring different models and evaluating their performance. This approach allows the team to research available options and then decide on the best model, rather than committing to the first one and only trying to improve it.

2.5. Will we have access to real rail break data or is it synthetic?

You'll be working with real-world production data provided by ARTC, including actual records of observed rail breaks and sensor readings from trains.

2.6. What format is the source data in, and where is it stored initially?

The data is initially stored in an SQL Server Database, and will be ingested into the Unity Catalog via the InsightFactory platform.

2.7. What kind of preprocessing is expected before training?

Preprocessing will involve: Feature selection, Feature engineering.

The kickoff meeting was productive and helped align everyone's expectations for the project. Following the discussion, the team agreed on several management practices: submitting sprint agendas in advance, sharing meeting notes regularly, and setting up a Teams group chat for smoother communication.

On the technical side, task tracking will be handled through a GitHub project board. The system's architecture is structured around three core layers: data ingestion, data processing, and machine learning.

Since the meeting, the team has created the Teams group, launched the GitHub board, and assigned initial tasks. The current focus is getting familiar with the platform, handling data preprocessing, working on feature engineering, and testing out various machine learning models.

To improve the next retrospective, the team plans to share progress updates beforehand and encourage more active participation.Kickoff meeting reflection.

3. Users

3.1. User 1: Machine Learning Engineer

Machine learning engineers are responsible for model development and optimization. They apply data ingestion for exploratory data analysis, preprocessing, and validation to develop and optimize models. They also conduct feature engineering and feature selection techniques to handle imbalanced temporal datasets, improving model accuracy and reliability.

3.2. User 2 : Operations Manager

As an Operations Manager, I want a weekly risk report and a ranked list of risky track segments, so that I can fix the worst ones first and plan budget and staff. I read short summaries of risk, delay minutes saved, and avoided failures, compare lines and months to see trends, and set simple goals for the model, so I can decide when we can use it more widely.

3.3. User 3: Safety Manager

As a Safety Manager, I want monthly safety reports and clear audit trails so that I can see risk and actions. I check if slow orders and inspections were done on time and if they worked. When risk is high, I ask for extra checks or updates to the plan. This helps prevent incidents and keeps trains running safely.

4. Software Architecture

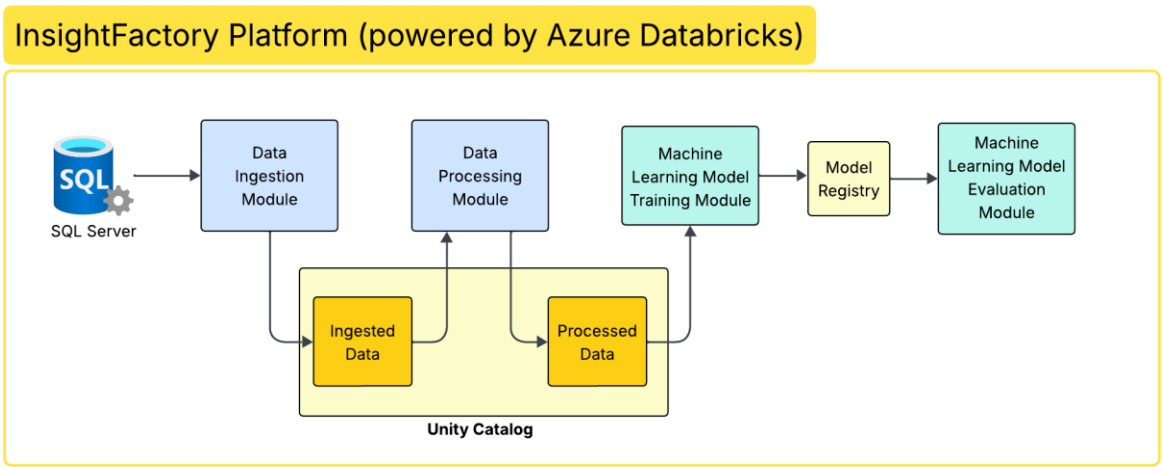


Figure 1. Software Architecture Diagram

The data pipeline will run on the InsightFactory platform, which supports us to organize modules, manage data, and execute code with cloud resources. Figure 1 illustrates the logical flow and interactions between system modules.

Raw data is stored in a SQL Server Database. The data ingestion module connects to the source data, extracts potential relevant data for processing, and stores it in Unity Catalog, which offers features such as data organisation, versioning, and access control.

The data processing module then processes the ingested data, performing feature selection to choose the most relevant and meaningful features to keep the model faster and simpler, and feature engineering to create additional useful features from data before model training. It may also perform data cleaning tasks, including handling missing values and anomalies.

Once processed, the data will be loaded into the model training module, which performs model selection, hyperparameter tuning, and training using the selected machine learning algorithms. Upon completion, the trained model will be submitted to the model registry. Finally, the evaluation module retrieves the registered model and tests it against a test dataset to assess its performance.

5. Tech Stack and Standards

5.1. Tech stack

- **Programming Languages:**
    - Python: The main language for data capture, data cleaning, feature extraction, and developing machine learning models.
    - PySpark: Handles large-scale data processing on Databricks. It also allows distributed computation for ARTC datasets.
    - SQL: Used for data extraction, transformation, and aggregation stored in Databricks Delta Tables.
- **Model Training & Machine Learning Frameworks:**
    - TensorFlow / Keras: Open-source frameworks for building, training, and deploying deep learning models.
    - PyTorch: A flexible framework that helps us experiment with different neural network architectures for detecting rail breakages.

    - Scikit-learn: Used for building baseline models, preprocessing data, and setting up evaluation pipelines to compare different machine learning approaches.
    - MLflow: For experiment tracking, model registry, and reproducibility within the Databricks environment.
- **Data Pipeline Engineering Platforms：**
    - InsightFactory.ai: Orchestrates ETL workflows, machine learning model training, and leaderboard submission in a cloud environment.
    - Unity Catalog & Delta Tables: Provides centralized data governance, ACID-compliant storage, and version control for datasets.

5.2. Agree on tools for communication and development

- **Microsoft Teams**: Primary platform for team meetings, file sharing, and discussions.
- **GitHub Project Board**: Task and issue tracking for collaborative development.
- **Databricks Notebooks**: Collaborative cloud-based coding and data exploration environment.
- **VSCode**: IDE for local Python/SQL development and version control integration.

5.3. Agree on coding standards

- **Python**:
    - Follow the PEP 8 style guide, including naming conventions, indentation rules, and comment standards.
    - Stick to concise coding practices. For example, keep each function under 30 lines to make it easier to read and maintain.
    - Use pylint for static code analysis to find potential errors and enforce coding standards.
- **SQL**:
    - Use upper-case for SQL keywords, such as SELECT, FROM, and WHERE.
    - Use snake_case for table and column names, like rail_break_locations and section_break_start_km.
    - Field names must have clear and descriptive meanings.
    - Considering query performance when writing SQL:
        - Minimize unnecessary joins.

        - Avoid overly complex queries where possible; break them into simpler, more maintainable parts.
    - Prevent critical mistakes:
        - Avoid running destructive operations like DROP TABLE or large-scale UPDATE without WHERE conditions.
        - Review and test queries in a test environment before executing on production datasets.
- **Version Control**:
    - Make all changes through Git feature branches.
    - Pull Requests must be reviewed by at least one team member before merging into the main branch.
    - Each commit must include a clear and descriptive commit message, explaining the purpose, scope, and changes made.
    - Different features and development iterations should use separate branches, with branch names that clearly reflect the iteration name and task being worked on, such as feature/v1-data-cleaning or bugfix/v2-model-evaluation.
    - Branch naming should follow a consistent convention to ensure traceability and maintainability.
- **Notebook Organization:**
    - Data ingestion & cleaning
    - Exploratory data analysis (EDA)
    - Feature engineering
    - Model training & evaluation
    - Result submission

5.4. Justifications for the choices

- **Python**: Python is widely used in machine learning and data engineering. It has many libraries for data processing and modeling.
- **PySpark**: PySpark handles distributed processing for large-scale datasets efficiently. It has integrated machine learning libraries (MLlib) that allow scalable, distributed model training directly on big data. This minimizes data movement and lets the same environment handle both preprocessing and training effectively.

- **InsightFactory.ai on Azure Databricks**: Combining data ingestion, transformation, and model training pipelines. It uses fully cloud-based resources and elastic scaling. There is no need for local installation of deep learning frameworks like PyTorch or TensorFlow, which boosts development efficiency and simplifies resource management, reducing setup time.
- **Microsoft Teams & GitHub Project Board**: Ensures efficient communication and project tracking in a distributed team.
- **PEP 8 & Version Control Standards**: Maintains code quality, readability, and reproducibility in a collaborative environment.

6. Group Meetings and Team Member Roles

6.1. How frequently and for how long do you intend to (virtually) meet within your group?

We are planning to hold group meetings at least once a week. Each group meeting will last for an hour. Additional meetings may be arranged if urgent issues arise.

6.2. What preparations should the group plan for before retrospective meetings?

Summarise and review: Summarise the work outcomes from the previous retrospective meeting to now, review progress on current action items, and track the status of ongoing tasks to ensure alignment.

Set clear goals: Outline the key points to be discussed at the next retrospective meeting, focusing on areas that are going well, areas that are not going well and need to improve next.

Draft agenda: Prepare an agenda for the upcoming meetings in accordance with the review and objectives to ensure that it reflects project progress and areas of concern.

Collect relevant data: Collect support material such as metrics, user feedback and code review to guide the discussion.

Promote open communication: Share the draft agenda with the group, encourage members to propose other discussion topics, and ensure everyone feels comfortable.

6.3. How will the group communicate with the tutor?

Regular updates: Weekly progress reports (snapshot) will be provided from assignment submit summarizing completed tasks, ongoing work, and any blockers.

Scheduled meetings: Hold regular Scrum meetings on Microsoft team meetings each two weeks with the tutor to discuss progress and gather feedback.

Real-time communication: Use Microsoft team channels for quick issue resolution. Make sure unexpected questions will be solved quickly and share opinions with other teams if needed.

6.4. Please name the Scrum Masters for each sprint (each team member can only be the Scrum Master for up to one sprint, except specific circumstances approved by the tutor).

Scrum Masters will be changed for each two week

Scrum Masters of Sprint 1: Zi Lun Ma (a1915860)

Scrum Masters of Sprint 2: Tianhua Zhang (a1915934)

Scrum Masters of Sprint 3: Di Zhu (a1919727)

Scrum Masters of Sprint 4: Jinchao Yuan (a1936476--)

Scrum Masters of Sprint 5: Zihan Luo (a1916700)

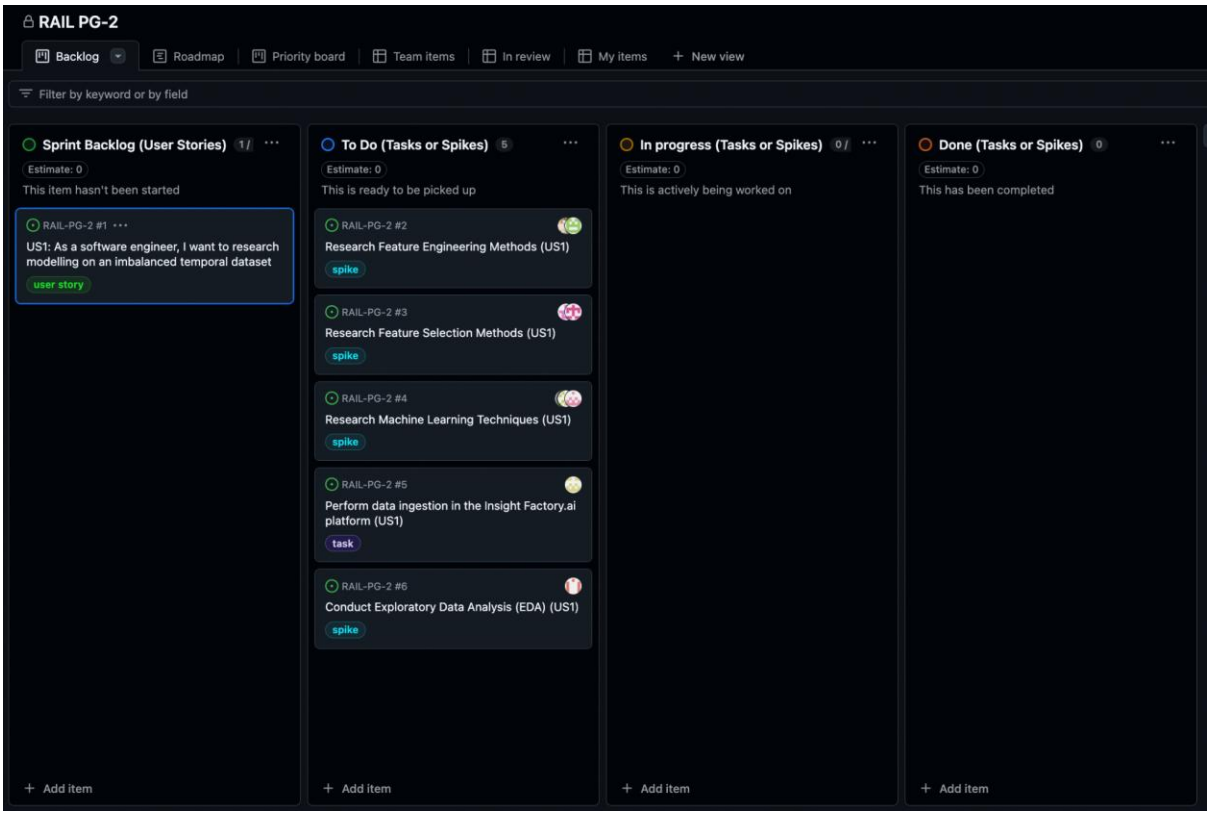
Snapshot

1. Product Backlog and Task Board

1.1. The product backlog

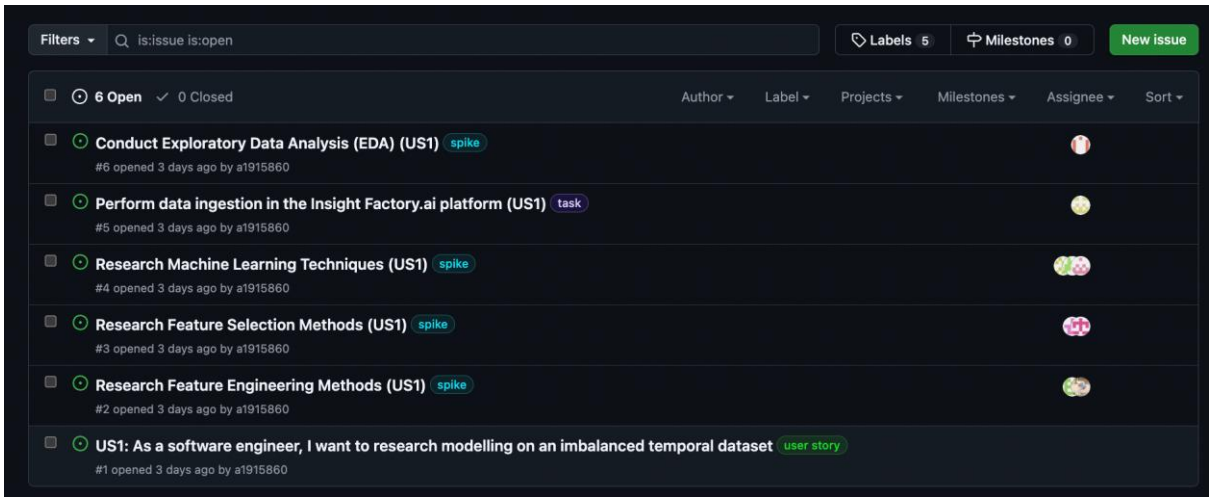
ID	Priority	User Story/Task/Spike	Description
PB1	1	Feature Engineering	Create new features based on domain knowledge and data patterns to improve model performance.
PB2	1	Feature Selection	Identify and retain the most relevant features to reduce noise and improve efficiency.
PB3	1	Model Research & Selection	Investigate suitable machine learning techniques for imbalance temporal datasets
PB4	2	Data Ingestion into InsightFactory.ai	Import the provided real-world production dataset into the InsightFactory platform.
PB5	2	Data Cleaning & Preprocessing	Handle missing values, outliers, and inconsistencies in the dataset.
PB6	2	Exploratory Data Analysis (EDA)	Analyze data distributions, trends, and anomalies to understand key characteristics.
PB7	3	Model Training	Train predictive models using the processed and engineered dataset.
PB8	3	Model Evaluation	Assess models using Accuracy, F1 Score, and AUCPR metrics.
PB9	3	Benchmark Comparison	Compare the model's performance against the InsightFactory bench mark model for potential bonus marks.
PB10	4	Model Optimization & FInalization	Fine-tune model parameters, optimize features, and prepare the final deliverable.

1.2. The task board



2. Sprint Backlog and User Stories

2.1. The Sprint backlog



2.2. User stories

Research techniques for modelling an imbalanced temporal dataset, including feature engineering, feature selection, and suitable machine learning methods. Ingest the data into the Insight Factory.ai platform. Conduct exploratory data analysis (EDA) to develop a plan to approach the project.

Related tasks:

1. Research Feature Engineering Methods
2. Research Feature Selection Methods
3. Research Machine Learning Techniques
4. Perform data ingestion in the Insight Factory.ai platform
5. Conduct Exploratory Data Analysis (EDA)

3. Definition of Done

A backlog item is considered "Done" when:
Spike:

- The research is complete, including findings, identified risks and challenges, and any recommendations.
- All relevant documentation is shared with the team.

Task*:

- Code (including database scripts) is implemented according to acceptance criteria.
- Code has been peer-reviewed and approved.
- All relevant tests (unit, integration) have been passed.
- Documentation (code comments, user guides) is updated.
- No major open defects remain.

* The current sprint is research-based, so the DOD for tasks might not be applicable. However, it's better to decide the expectation for tasks earlier.

4. Summary of Changes:

Since the start of the project, our team has analysed the provided user stories. Based on these, we have defined the key tasks for the first sprint.

Some of the team members will focus on researching feature engineering methods, feature selection methods, and machine learning techniques. After these are done, a report on findings and recommendations for implementation will be submitted.

The other members will perform data ingestion into the InsightFactory.ai platform, and then conduct exploratory data analysis (EDA) to understand the dataset's structure, distributions, and potential data quality issues.

In our opinion, this work will provide a solid foundation for developing the modelling approach in subsequent sprints.