Metro Train APU Anomaly Detection Application

Objective

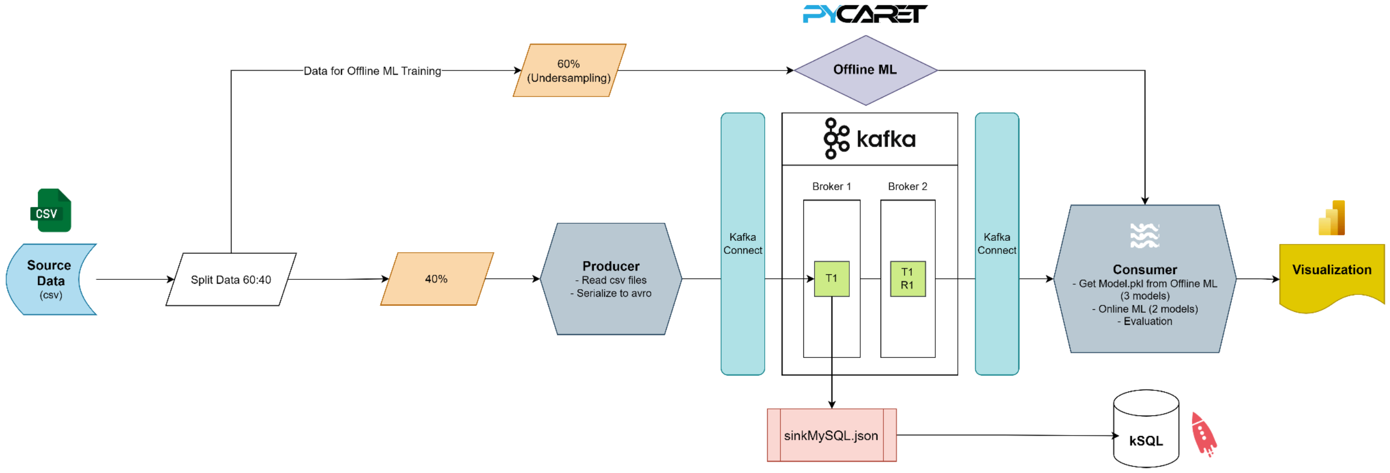
- To design and analyze data in both streaming data and historical data in order to detect the failure system of given data

Solution Idea

- Algorithms

When there is a system failure, *The System* will predict damage utilizing data from 15 signals and then provide a comparison of the predicted and actual results. *The Model* Algorithms were configured to present the predicted result with the highest accuracy score from five models: three offline models (Gradient Boosting Classifier, Light Gradient Boosting Machine, Extra Trees Classifier) and two online models (Hoeffding Tree Classifier and Extremely Fast Decision Tree Classifier). The output results will be shown on *the Monitoring Visualization Dashboard* with Power BI Online.

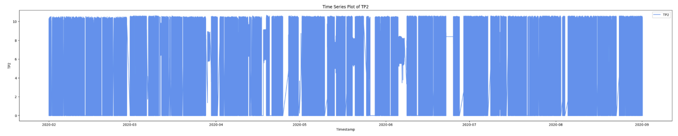
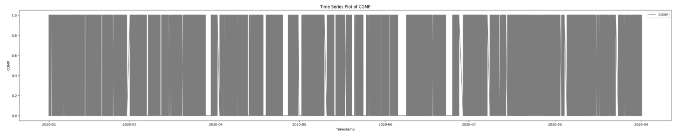
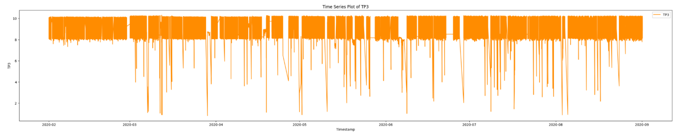
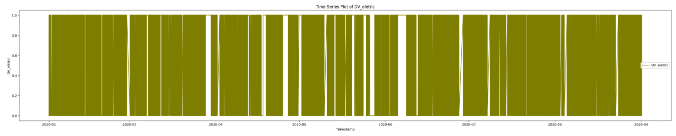
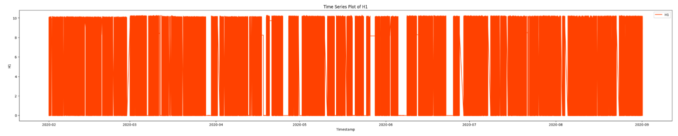
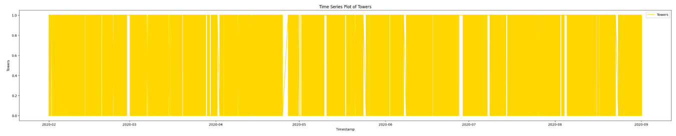
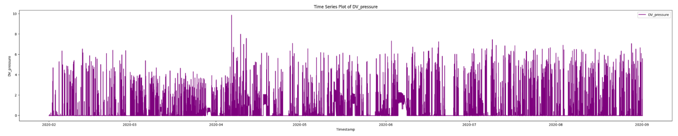
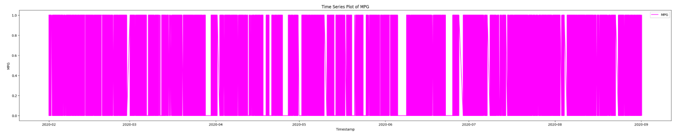
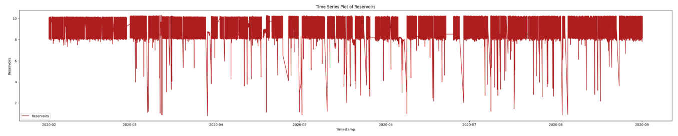
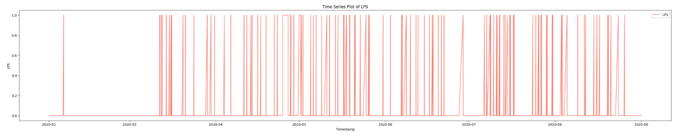
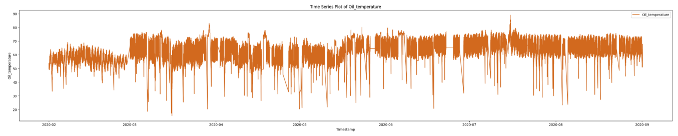
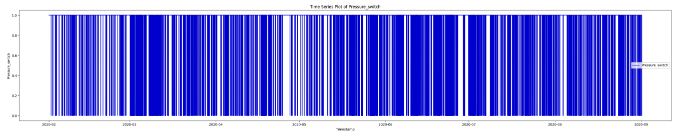
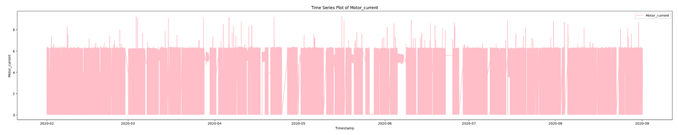
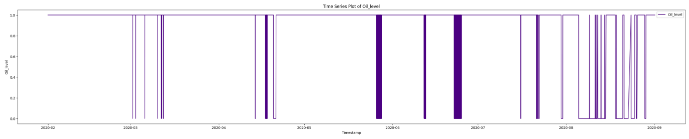
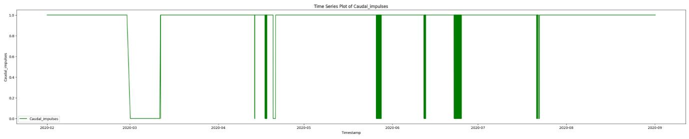
- Flowcharts

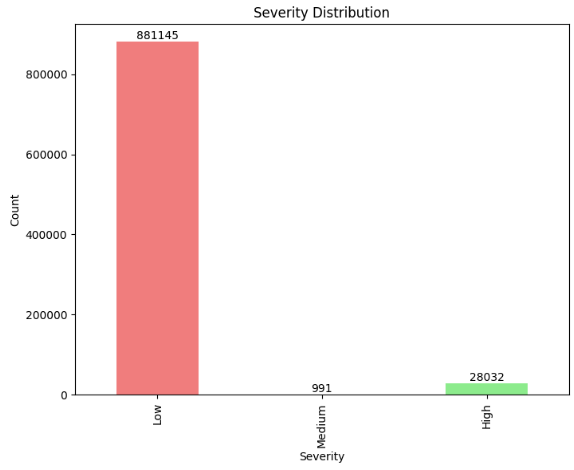
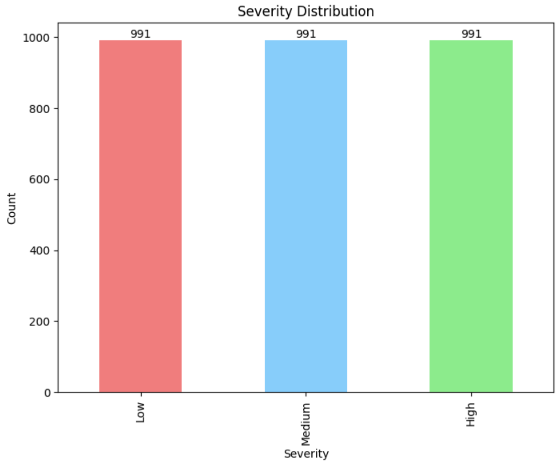


1. Get CSV file from the references source
2. **Split** the given data into 2 groups: 60% for training Offline ML and 40% for Online ML
   - First 60% data for Offline ML, found the imbalance data, thus we applied the Undersampling technique to resampling data for balanceness before apply *PyCaret* to predict the result of y (The Level of Severity)
   - The rest 40% of data will go to streaming simulation for Online Model Prediction
3. **Producer** was assigned to read the 40% of raw data with csv format and serialized the data as avro format in order to convey the data via kafka connect and store to the topic 1 (rawData)
4. In order to apply the ability of fault tolerance, and increase the system performance, the **Kafka system** was designed to contain *two brokers, one topic, two partitions, and two replicas*.
5. **Consumer** was in charge of calling Offline ML Models via .pkl files, predicting Online ML Models utilizing *the River ML library*, developing ensemble algorithms with *the scikit-learn library*, and embedding the *Power BI API* for streaming visualization.
6. In the parallel task, the **sink connector** had been assigned with reading the data from topic 1 (rawData) and writing the raw data to kSQL to store the log data from the streaming line.
7. **Power BI** was designed to display the predicted result (after ensemble selection) compared to actual result. Furthermore, it monitors data flows by monitoring the table and trigger chart to determine whether data was received.

Experiment:

EDA

Devices	Censors
	
	
	
	
	
	
	
	

Imbalance Data	Resampling Data
	

Train and Test Process

Offline ML

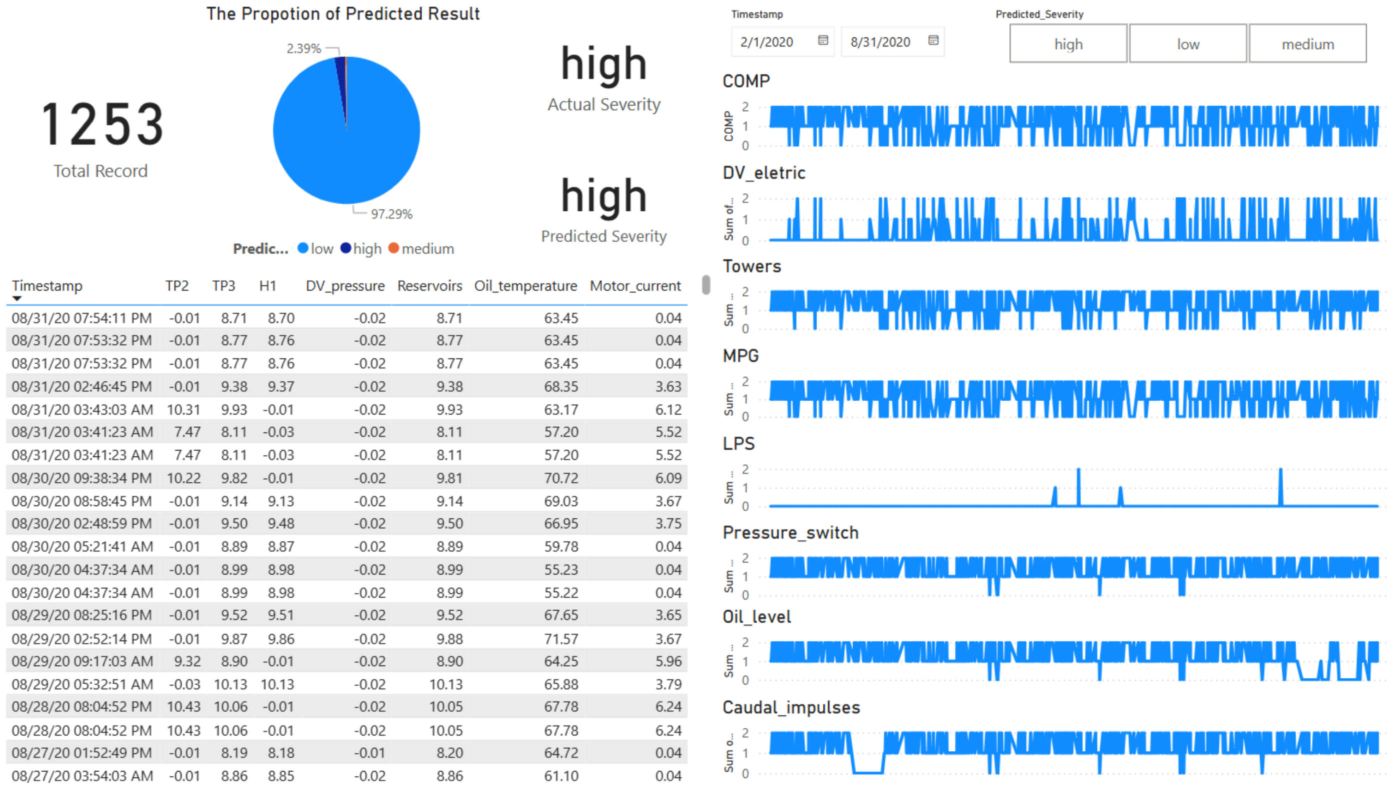
1. Data Preparation:
    - Drop unused features (e.g. Unnamed: 0)
    - The timestamp was set as the index
    - The target variable (severity) was found to be imbalance
2. Resampling:
    - Undersampling was applied to balance the data, cutting out the over-represented classes
3. Model Training:
    - The resampled data was used for training offline ML models
    - PyCaret library was utilized to implement classification models
    - The three models with the highest accuracy scores were selected: Gradient Boosting Classifier, Light Gradient Boosting Machine, Extra Trees Classifier

Online ML

1. Data Streaming Simulation
    - The remaining 40% of the data was used for streaming simulation and online ML training and prediction
    - The River ML was utilized to implement classification models
    - The two models with the classification methods were selected: Hoeffding Tree Classifier and Extremely Fast Decision Tree Classifier

Result

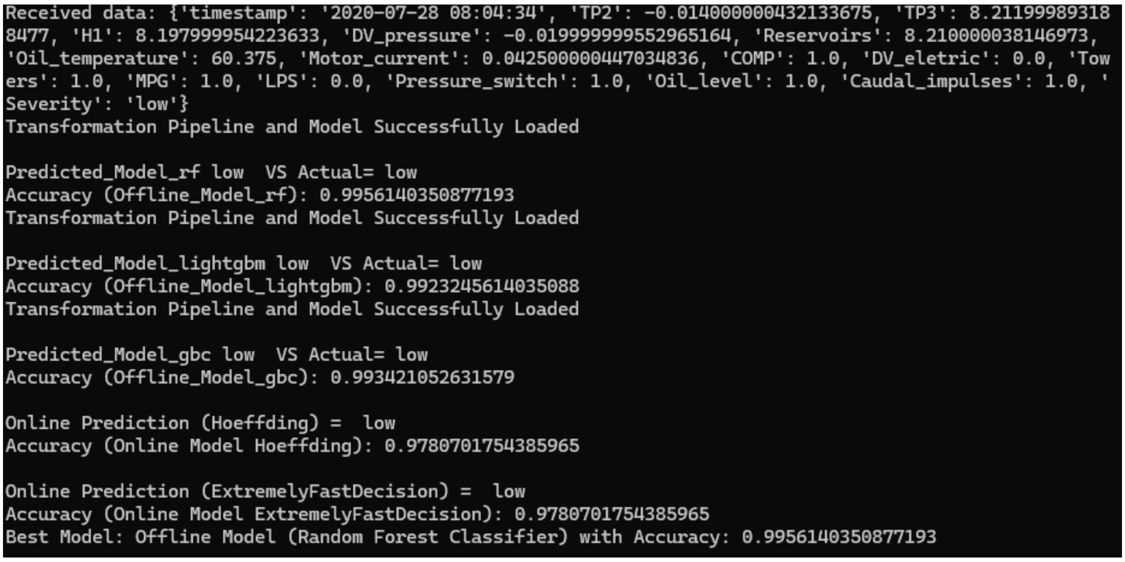
The anomaly detection system successfully integrated both offline and online machine learning models. The use of Kafka for data streaming and Power BI for real-time visualization provided an efficient solution for monitoring and predicting system failures. The models achieved high accuracy, and the system design ensured fault tolerance and high performance.



Timestamp	TP2	TP3	H1	DV_pressure	Reservoirs	Oil_temperature	Motor_current
08/31/20 07:54:11 PM	-0.01	8.71	8.70	-0.02	8.71	63.45	0.04
08/31/20 07:53:32 PM	-0.01	8.77	8.76	-0.02	8.77	63.45	0.04
08/31/20 07:53:32 PM	-0.01	8.77	8.76	-0.02	8.77	63.45	0.04
08/31/20 02:46:45 PM	-0.01	9.38	9.37	-0.02	9.38	68.35	3.63
08/31/20 03:43:03 AM	10.31	9.93	-0.01	-0.02	9.93	63.17	6.12
08/31/20 03:41:23 AM	7.47	8.11	-0.03	-0.02	8.11	57.20	5.52
08/31/20 03:41:23 AM	7.47	8.11	-0.03	-0.02	8.11	57.20	5.52
08/30/20 09:38:34 PM	10.22	9.82	-0.01	-0.02	9.81	70.72	6.09
08/30/20 08:58:45 PM	-0.01	9.14	9.13	-0.02	9.14	69.03	3.67
08/30/20 02:48:59 PM	-0.01	9.50	9.48	-0.02	9.50	66.95	3.75
08/30/20 05:21:41 AM	-0.01	8.89	8.87	-0.02	8.89	59.78	0.04
08/30/20 04:37:34 AM	-0.01	8.99	8.98	-0.02	8.99	55.23	0.04
08/30/20 04:37:34 AM	-0.01	8.99	8.98	-0.02	8.99	55.22	0.04
08/29/20 08:25:16 PM	-0.01	9.52	9.51	-0.02	9.52	67.65	3.65
08/29/20 02:52:14 PM	-0.01	9.87	9.86	-0.02	9.88	71.57	3.67
08/29/20 09:17:03 AM	9.32	8.90	-0.01	-0.02	8.90	64.25	5.96
08/29/20 05:32:51 AM	-0.03	10.13	10.13	-0.02	10.13	65.88	3.79
08/28/20 08:04:52 PM	10.43	10.06	-0.01	-0.02	10.05	67.78	6.24
08/28/20 08:04:52 PM	10.43	10.06	-0.01	-0.02	10.05	67.78	6.24
08/27/20 01:52:49 PM	-0.01	8.19	8.18	-0.01	8.20	64.72	0.04
08/27/20 03:54:03 AM	-0.01	8.86	8.85	-0.02	8.86	61.10	0.04

Error Metrics

The error metrics demonstrate that the models achieved high accuracy in predicting system failures. The online models also performed well, ensuring reliable real-time anomaly detection. These results validate the effectiveness of the implemented machine learning algorithms and the robustness of the system design for real-time monitoring and predictive maintenance. This evaluation confirms that the system is capable of accurately identifying anomalies and potential failures, providing a valuable tool for maintaining the operational integrity of the Metro Train APU system.



```
Received data: {'timestamp': '2020-07-28 08:04:34', 'TP2': -0.014000000432133675, 'TP3': 8.21199989318
8477, 'H1': 8.197999954223633, 'DV_pressure': -0.019999999552965164, 'Reservoirs': 8.210000038146973,
'Oil_temperature': 60.375, 'Motor_current': 0.042500000447034836, 'COMP': 1.0, 'DV_eletric': 0.0, 'Tow
ers': 1.0, 'MPG': 1.0, 'LPS': 0.0, 'Pressure_switch': 1.0, 'Oil_level': 1.0, 'Caudal_impulses': 1.0, '
Severity': 'low'}
Transformation Pipeline and Model Successfully Loaded

Predicted_Model_rf low  VS Actual= low
Accuracy (Offline_Model_rf): 0.9956140350877193
Transformation Pipeline and Model Successfully Loaded

Predicted_Model_lightgbm low  VS Actual= low
Accuracy (Offline_Model_lightgbm): 0.9923245614035088
Transformation Pipeline and Model Successfully Loaded

Predicted_Model_gbc low  VS Actual= low
Accuracy (Offline_Model_gbc): 0.993421052631579

Online Prediction (Hoeffding) =  low
Accuracy (Online Model Hoeffding): 0.9780701754385965

Online Prediction (ExtremelyFastDecision) =  low
Accuracy (Online Model ExtremelyFastDecision): 0.9780701754385965
Best Model: Offline Model (Random Forest Classifier) with Accuracy: 0.9956140350877193
```

Summary and Suggestions

The Metro Train APU Anomaly Detection Application effectively integrates both offline and online machine learning models to detect system failures. The system has shown high accuracy in predicting anomalies, validating the robustness of the design and the effectiveness of the algorithms used. This makes it a valuable tool for real-time monitoring and predictive maintenance, helping maintain the operational integrity of the Metro Train APU system.

Suggestion for Improvement

- **Feature Engineering:** Explore feature engineering to identify additional features and employ advanced feature selection techniques to pinpoint the most relevant ones.
- **Model Algorithms and Optimization:** Experiment with sophisticated algorithms and conduct hyperparameter tuning to achieve optimal performance as well as adopt a more suitable model for anomaly detection.
- **Data Augmentation:** Use synthetic data generation to balance the dataset, especially for rare events and underrepresented classes.
- **System Scalability:** Implement distributed computing frameworks to handle larger datasets efficiently.
- **Real-Time Analytics:** Integrate advanced visualization tools and *real-time alerts system*
- **User Feedback:** Collect feedback from end-users and conduct training sessions to enhance the usability and functionality of the monitoring dashboard.
- **Continuous Monitoring and Maintenance:** Implement continuous monitoring of model performance and schedule regular updates to maintain accuracy and incorporate the latest advancements in machine learning.
- **Refinement and Enhancement:** Follow these suggestions to further refine the Metro Train APU Anomaly Detection Application, ensuring reliable and accurate predictions for smooth and safe metro train operations.