EAT40005 Engineering Technology Projects A

Individual Work Log

PROJECT NAME:	OBD-II Based Predictive Maintenance System		
STUDENT NAME:	**Dang Khoa Le**		
STUDENT ID:	**103844421**	**WEEK # (& dates covered):**	**#9**

TASKS	STATUS	TIME SPENT	ACTION ITEM/NOTE
Task	**Status**	**Time Spent**	**Action Item/Note**
Successfully collected physical OBD-II data using the ELM327 adapter across 4 driving trials.	Completed	0.5 hours	Dale drove his car, and I collect it.
Loaded, explored, and analyzed the collected data in Google Colab.	Completed	1 hours	Break down the shape, feature and insights from the 4 trials in CSV files.
Merged datasets based on common columns only, resulting in a consolidated dataset.	Completed	1 hour	Enlarge the dataset.
Implemented a comprehensive data cleaning pipeline	Completed	3 hours	Duplicate row & column removal. Constant-value and empty-column detection. Placeholder error filtering (255, -40, -22). Row-wise missing threshold filtering. Timestamp parsing and missing detection. Outlier capping (RPM outside [100, 6000]). MinMax normalization.
Designed and applied basic feature engineering	Completed	2 hours	Average engine load (AVG_ENGINE_LOAD). Aggregated temperature (TEMP_MEAN). Derived metric for air efficiency (AIRFLOW_PER_RPM).
Generated visual insights	Completed	1 hours	Correlation heatmap. Time-series trends for key signals (RPM, Load, Temp). Missing timestamp detection through time delta plotting.
Improved the OBD-II Python Raspberry Pi logging script	Completed	0.5 hour	Split polling of high- and low-frequency PIDs. Robust WiFi and serial failover handling. Dynamic CSV logging with initial full PID snapshot.
Delivered a well-structured 9-slide presentation to demonstrate physical data logging, cleaning pipeline, and Python logger implementation	Completed	2 hour	
TOTAL WEEKLY TIME SPENT		**11 hours**	

TASKS PLANNED FOR NEXT WEEK	EXPECTED COMPLETION
Conduct longer data logging sessions over multiple driving scenarios.	Week 10
Implement timestamp-based resampling for fixed-Hz ML model preparation.	Week 10
Begin state labeling strategy based on speed/RPM thresholds.	Week 10
Draft anomaly detection prototypes using early models (XGBoost, Autoencoder).	Week 10-11
Group/remove high correlated features	Week 10-11
Do Research Report	Week 10

Summary/weekly reflection for Week 9:

- **Key Tasks Done:**

◊ Successfully collected physical OBD-II data using the ELM327 adapter across 3 real-world driving trials.
◊ Loaded, explored, and analyzed the collected data (test1.csv, test2.csv, test3.csv) in Google Colab.
◊ Merged datasets based on common columns only, resulting in a consolidated dataset.
◊ Implemented a comprehensive data cleaning pipeline, including:
- Duplicate row & column removal.
- Constant-value and empty-column detection.
- Placeholder error filtering (e.g., 255, -40, -22).
- Row-wise missing threshold filtering.
- Timestamp parsing and missing detection.
- Outlier capping (e.g., for RPM outside [100, 6000]).
- MinMax normalization.

◊ Designed and applied basic feature engineering, including:
- Average engine load (AVG_ENGINE_LOAD).
- Aggregated temperature (TEMP_MEAN).
- Derived metric for air efficiency (AIRFLOW_PER_RPM).

◊ Generated visual insights:
- Correlation heatmap.
- Time-series trends for key signals (RPM, Load, Temp).
- Missing timestamp detection through time delta plotting.

◊ Improved the OBD-II Python Raspberry Pi logging script, allowing:
- Split polling of high- and low-frequency PIDs.
- Robust WiFi and serial failover handling.
- Dynamic CSV logging with initial full PID snapshot.

◊ Delivered a well-structured 8-slide presentation to demonstrate physical data logging, cleaning pipeline, and Python logger implementation.

- **Key Learning:**

◊ Cleaning real-world OBD-II data is drastically different from public datasets due to:
  - Inconsistent sensor availability.
  - Connection delays and packet loss.
  - Sensor saturation and encoding quirks.

◊ Preventing data leakage and bias starts at cleaning — filling all missing values with median may distort actual driving behavior if done blindly.
◊ Logger script enhancements (e.g., PID grouping, fallback polling) are crucial for efficiency.
◊ Insightful visualizations allow us to uncover sensor drift, inactivity, or logging issues early on.
◊ Future work will focus on labeling driving states (idle, low speed, high speed) for predictive modeling and supervised training opportunities.

- **Literature/Resources Reviewed:**

Understanding and Knowledge taken from last few weeks research works.

- **Issues Faced:**

◊ Low sampling rate (~0.07 Hz) due to sequential PID queries (~200 ms per PID).
◊ Logging resulted in large gaps between readings, requiring post-processing strategies.
◊ Some trials recorded only timestamps and missing sensor values due to temporary loss of connection or data lockout during startup phase.
◊ Removing near-empty rows and duplicated placeholders required conditional logic.
◊ Timestamp axis in plots introduced unwanted gaps – addressed by switching to uniform integer-based indexing for clearer visualization.