Assignment 1: Develop Supervised Machine Learning Models – Report

Prashanth Vangari - 11645119

Task 1: Data Preparation

1. Opening and Reading Files: We are passing the three log files to read and store it in the variables. **In Google Colab, we need to upload the files(logs) every time we restart the google colab.**
2. Data Processing: Here we are initializing empty lists for 'speed' and 'rpm' to store specific data. It iterates through each line in the 'read_file' and processes the data within each line.
3. Speed and RPM Processing:
    a. Based on the 'PID' value, the code differentiates between speed and RPM data.
    b. If the 'PID' is '254' (indicating speed data), it checks if the 'value' is greater than or equal to 4095. If so, it sets 'attack' to 1 and adjusts the 'value' accordingly.
    c. If the 'PID' is '115' (indicating RPM data), it checks if the 'value' is greater than or equal to 65535. If so, it sets 'attack' to 1 and adjusts the 'value' accordingly.
    d. Processed 'value' data is appended to the respective 'speed' or 'rpm' lists.
4. DataFrame Conversion: After processing all lines, the code converts the 'canData' list of dictionaries into a Pandas DataFrame.
5. Data Cleaning: Unnecessary columns ('stamp,' 'const1,' 'change,' and 'value2') are dropped from the DataFrame. Rows in the DataFrame are filtered to include only 'PID' values of '115' (RPM) or '254' (Speed).
6. OneHot Encoding: Onehot encoding is applied to the 'PID' column to convert categorical data into numerical form. The 'PID' column is dropped, and the onehot encoded columns are added to the DataFrame.

Task 2: Explore and Analyze the Data

In the task two there are three steps.

Step1. Scatter Plots for Speed and RPM Over Time:

Separate scatter plots are created for each scenario (FFF Injection, RPM Injection, No Injection). The code generates scatter plots showing how Speed and RPM change over time for each scenario. These plots help visualize the patterns and fluctuations in Speed and RPM readings.

Step2 : Frequency plot

Frequency plots are generated for each scenario to understand the distribution of Speed and RPM values over time. These plots show the frequency or occurrence of different Speed and RPM values.

Observations: After generating the plots, the observations are given based on the visualizations.

Step 3: Corelation and P-values:

2. The Pearson correlation coefficients in Task 2 measure the strength and direction of the linear relationships between Speed and RPM in different scenarios (FFF Injection, RPM Injection, No Injection).
3. P-values: If the p-value is close to 0 then it indicates that the there is very less chance of getting these values by chance.

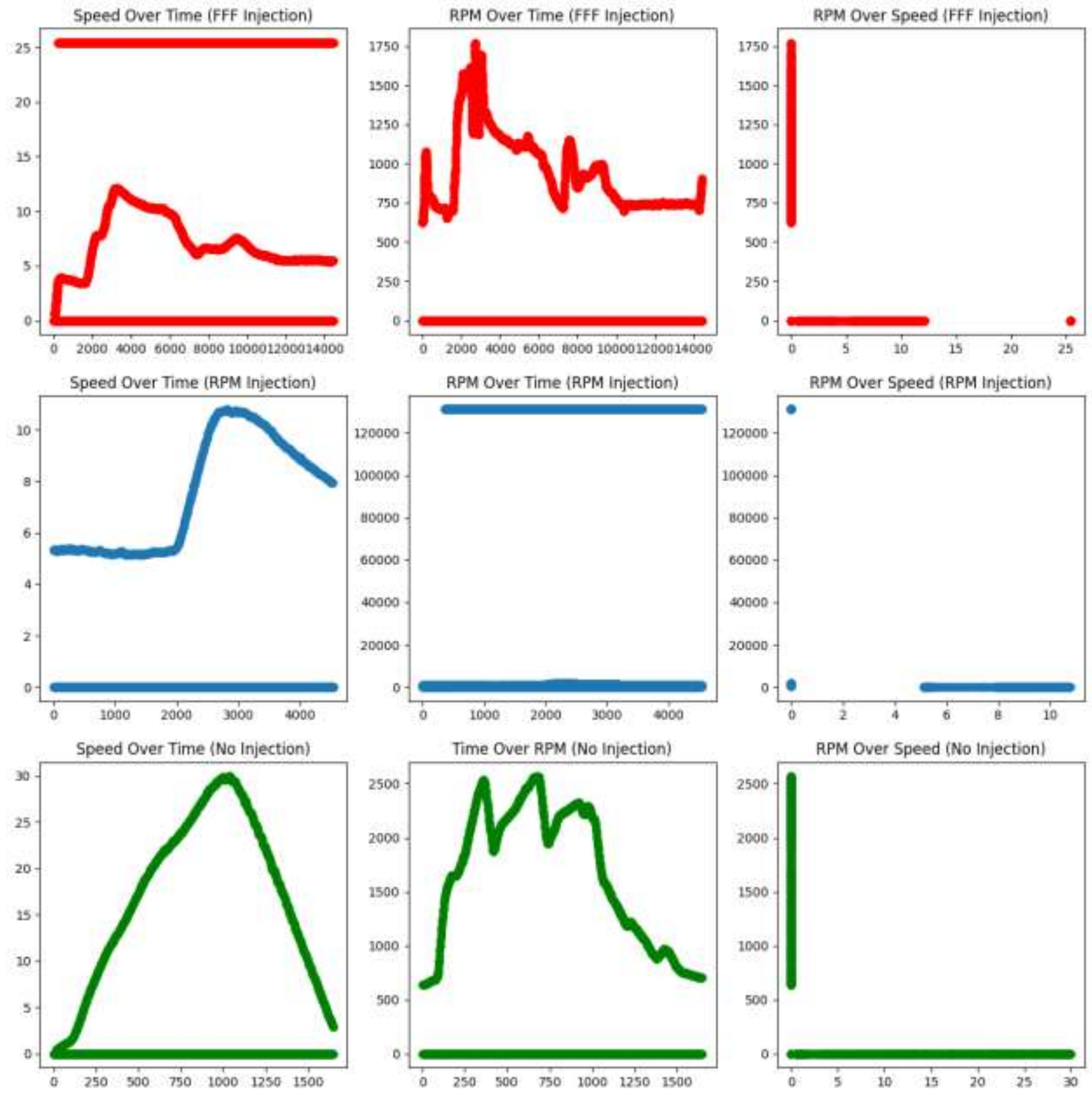

Fig 1. 9 Scatter plots- 3 for each data frame (Removed from the code as it was not asked)

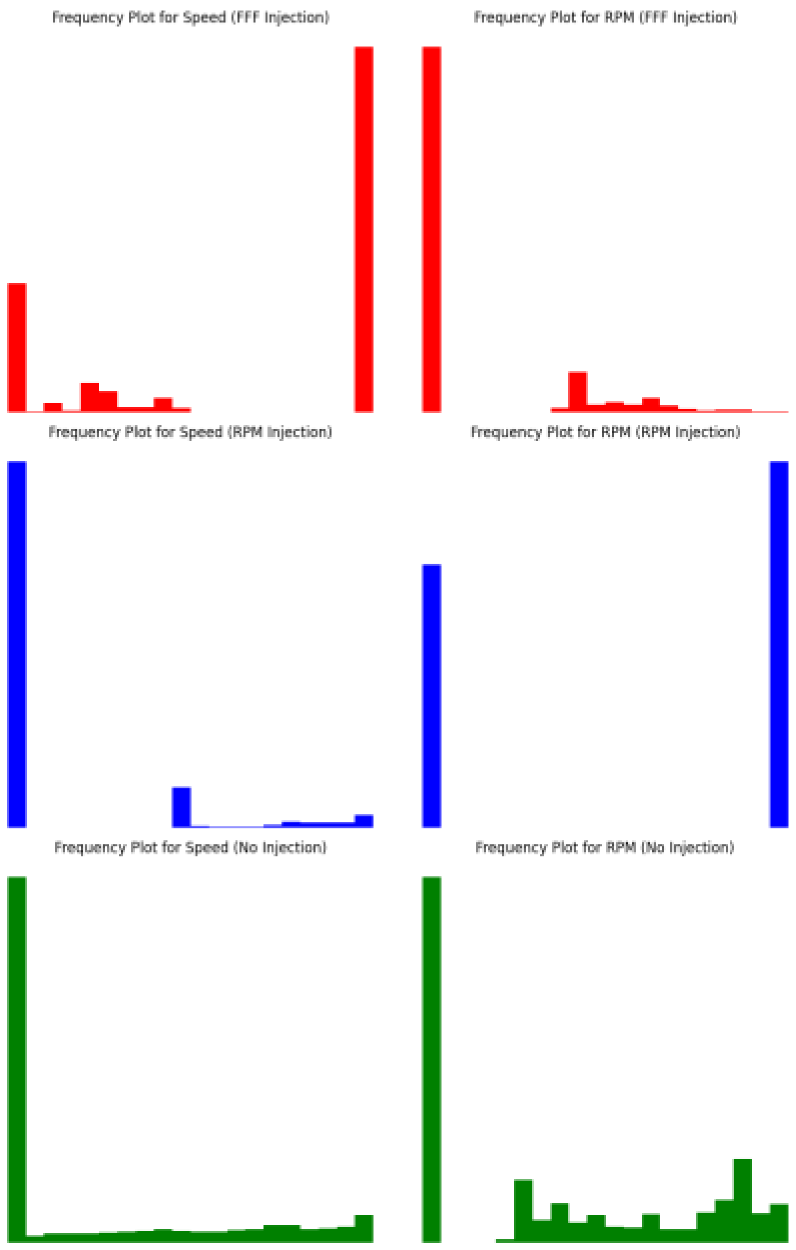

Fig 2: 6 Histograms – 2 for each data frame.(Removed from the code as it was not asked)

Task 3: Supervised Machine Learning Model

This section involved building a supervised machine learning model, specifically a Random Forest Classifier, to detect potential attacks based on speed and RPM data. The data was split into training and testing sets, and the model was trained on the training data. Subsequently, predictions were made on the test data, and performance metrics, including a confusion matrix and classification report, were presented to evaluate the model's effectiveness in attack detection.

1. **Combining Data:** The data from three different data frames (representing different scenarios: FFF Injection, RPM Injection, and No Injection) are combined into a single dataset.

2. Data Splitting: The combined dataset is split into training set and a test set. 75% of the data is used for training, and the 25% is used for testing.

3. Feature Selection: The input features (X) and the target variable (y) are defined. In this case, 'Speed' and 'RPM' are selected as the input features, and 'attack' is chosen as the target variable.

4. Model Creation: A Random Forest Classifier is created and initialized with a specified random seed (random_state=42).

5. Model Training: The Random Forest Classifier is trained using the training data (X_train and y_train).

6. Prediction: The trained model is used to make predictions on the test dataset.

7. Confusion Matrix: A confusion matrix is used to evaluate the model's performance. It represents the number of true positives, true negatives, false positives, and false negatives.

8. **Classification Report:** A classification report is generated, which provides various metrics such as precision, recall, F1-score.

Task 4: Second Choice of ML Model

In this section, an alternative machine learning model, Logistic Regression, was implemented for attack detection. Similar to Task 3, the data was split, the model was trained, and predictions were generated and evaluated using a confusion matrix and classification report.

Task 5: Comparison of Models

The performance of the two machine learning models (Random Forest Classifier and Logistic Regression) was compared based on the confusion matrices. Both models showed perfect true positives and true negatives. **However, Logistic Regression showed a small number of false positives, which means it identified an attack even when there was none.** Since the remaining evaluation metrics are same and the logistic regression has a small false positive, **a conclusion was drawn that Random Forest is performing better than the logistic regression.**

Task 6: Discussion

This section discussed the overall findings and challenges encountered during the analysis. One of the primary challenges was dealing with imbalanced data, where the number of non-attacks significantly exceeded attacks. Addressing this issue was crucial to prevent model bias. Additionally, it was noted that supervised machine learning may not always be the best approach for anomaly detection, and alternative methods such as anomaly detection algorithms were suggested.