Eagle ML - Dataset Analysis Report

Name: Chaitanyavishnu Gadde
Contact: cvsgadde@gmail.com

1. Introduction:

This report outlines how I handle, analyze, and draw conclusions from our data to make informed choices.

2. Data Acquisition: Merging Multiple Data Sources

The analysis was rooted in data sourced from three primary tables:

- **Sensor.csv:** A product of the OSIPi system, this dataset held time-series data across 27 columns. Columns with the prefix "B_" were sourced directly from sensors. The 'Cycle ID' was a unique identifier for every cycle, and 'timestamp' indicated each cycle's completion time.
- **Sensor_high_freq.csv:** This high-frequency data, derived from the controller, offered more granular insights with columns prefixed by "B_". The "Percent" column depicts the carrier occupation percentage, which is crucial for understanding system efficiency.
- **Percent_reference.csv:** A pivotal addition to the data arsenal, this dataset wasn't a time series. Instead, it offered combustion-related data, providing recipes corresponding to different carrier percentages. This information was crucial in understanding the nuances of how combustion varied with carrier percentage and provided insights into optimal combustion recipes.

3. Data Cleaning and Preprocessing

1. **Handling Missing Data:**
   First, I identified vital columns. Any row lacking data in these columns was removed. For the rest, missing spots were filled with medians.

2. **Target Variable Processing:**
   The 'Good/Bad' column, indicating cycle quality, was cleaned up for clarity, ensuring a clear good (1) or bad (0) label.

3. **Non-Numeric Data Handling:**
   I converted numbers that were in string format into numbers. If there were any issues, I filled the gaps using the column’s median value.

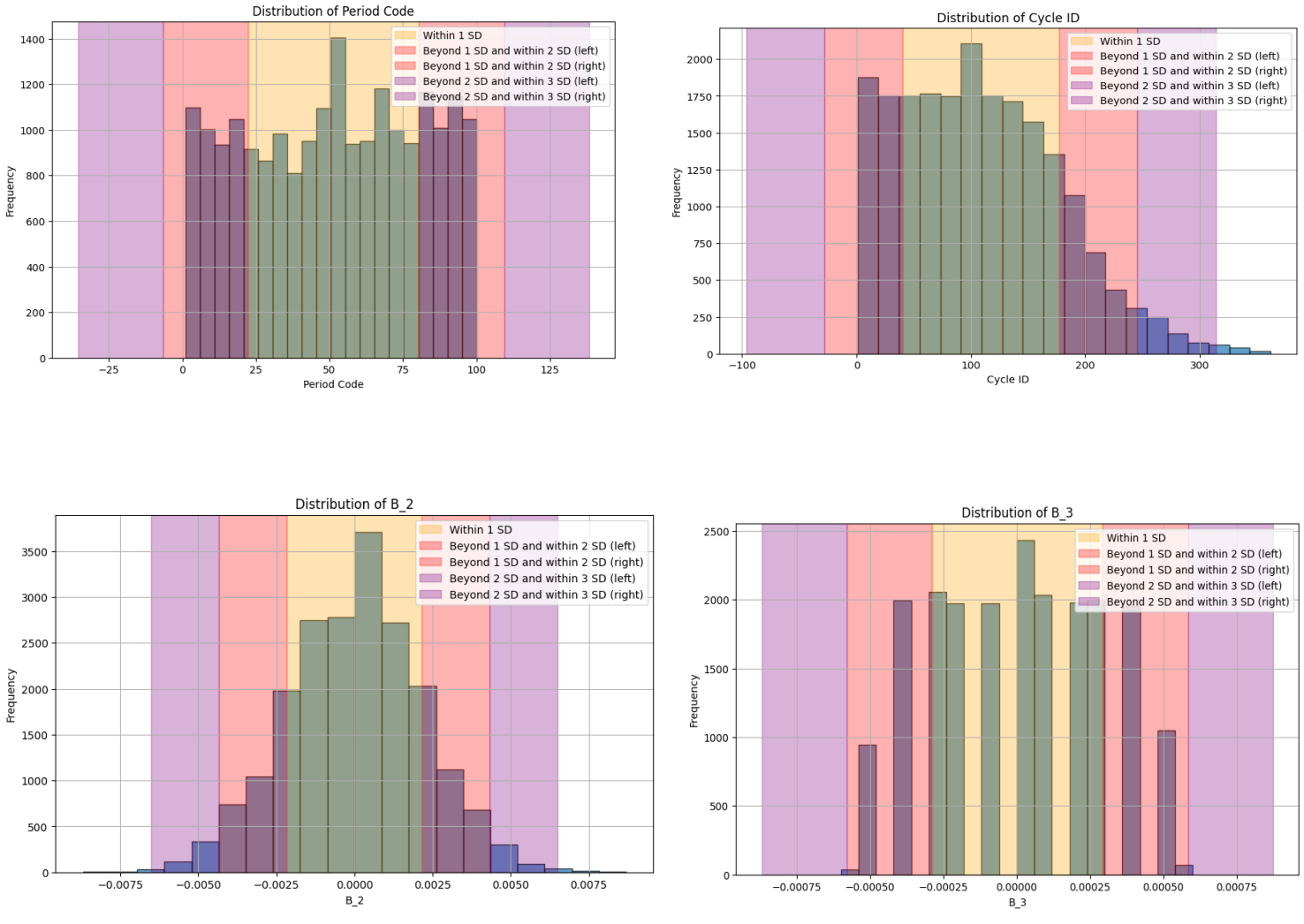
4. Detecting and Removing Outliers

1. **Using Standard Deviation:**
   I marked values that were too far from the average (more than three standard deviations) as outliers.

2. **Visual Checks:**
   I visualized data distributions, marking regions at 1, 2, and 3 standard deviations to identify and confirm outliers.

5. Visualization of Data Distributions

I drew charts to show how our data spreads, using standard deviation markings to understand the spread and spot outliers.



Figures: Visuals of some of the columns, showing 1,2 and 3 standard deviations

6. Feature Selection

I had a lot of features in my data, like 'B_2', 'B_3', 'B_4', and others. To make things more straightforward, I removed some of the details that didn't seem too important. I used a random forest classifier to figure out which features were relevant and which weren't. By doing this, I made sure that when I trained the models, they were learning from only the most important information.

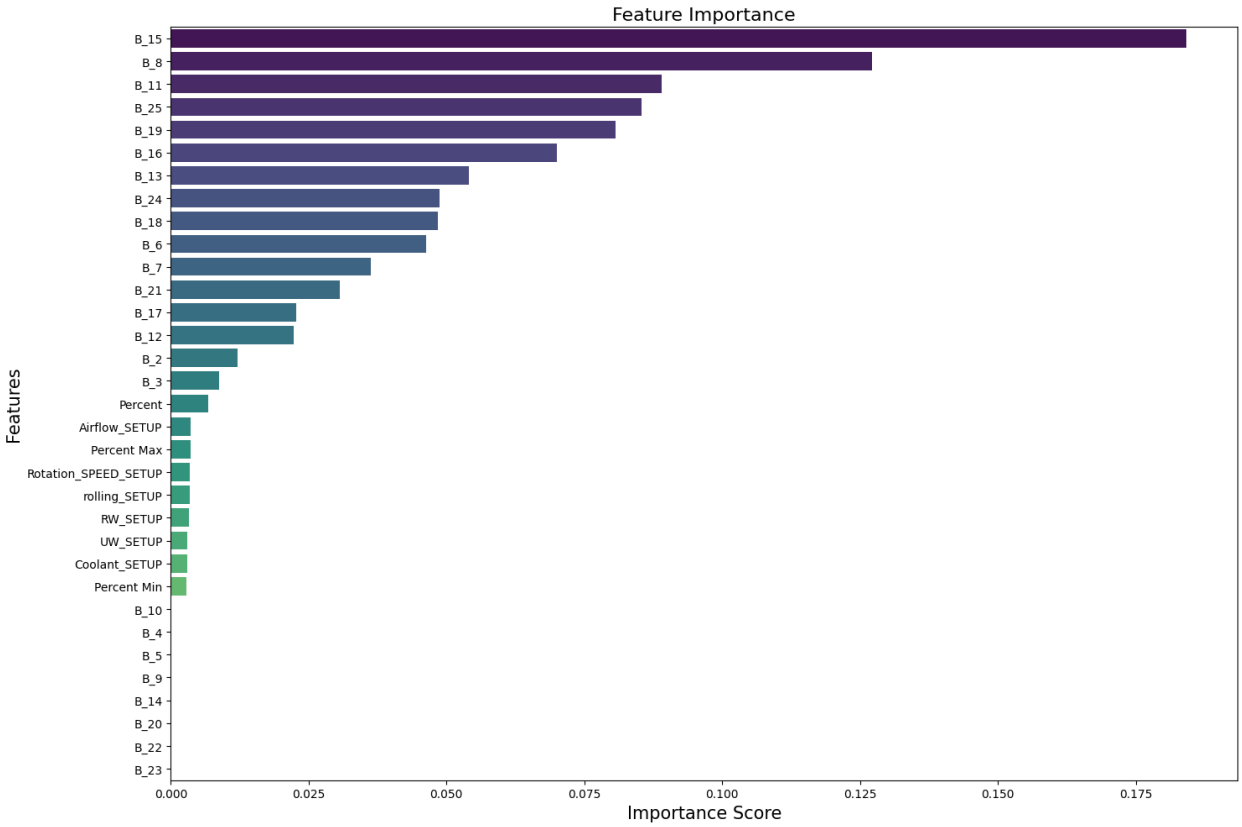


Figure: Feature Importance Scores

7. Addressing Class Imbalance with SMOTE

I noticed that my data had more of one type of class than the other. This can make computer models favor the one that appears more often. To fix this, I used a method called SMOTE, or Synthetic Minority Over-sampling Technique. With SMOTE, I added some new data points for the class that had fewer examples. By doing this, I made sure my classification models learned from a balanced set of data, which helped in making fair and correct predictions.

8. Model Training and Evaluation

- **Random Forest:**
  I used an ensemble model called Random Forest because it's good at looking at a lot of data and figuring out which details are most important. It did a great job and was accurate about 97.67% of the time.

- **XGBoost:**
  Then, I tried another ensemble model called XGBoost. It's special because it tries to learn from any mistakes it made in the past. It did even better and was correct about 98.29% of the time.
- **LSTM (Long Short-Term Memory):**
  Lastly, I used a model called LSTM. It's good at spotting patterns over time. After training it for a while (50 epochs, to be exact), it got things right about 95.90% of the time.

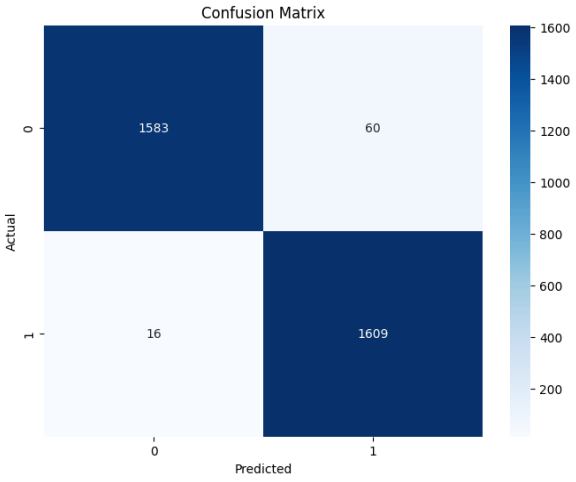


Figure: Confusion Matrix of Random Forest Classifier

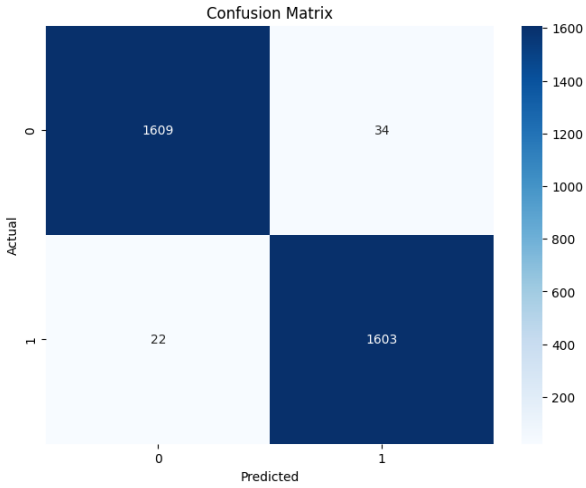


Figure: Confusion Matrix of XGBoost Classifier

9. K-Fold Cross-Validation and Ensemble Modeling

I made sure to test my models really well. I split my data into 5 parts and trained the models 5 times, each time using a different part as a test. This method is called K-Fold Cross-Validation. After these tests, I noticed that each model had its own strengths. So, instead of relying on just one model, I combined the best ones together. This is called "model ensembling". By using them all at once, their strengths came together, and I could get even better and more accurate predictions.

10. Deploying on Azure

I developed an application that integrates a Flask backend to facilitate smooth user interactions. By leveraging APIs within the app, users can effortlessly upload test files, making the process of acquiring predictions and metrics both intuitive and seamless. Recognizing the need for wide-reaching availability and steadfast performance, I decided to host the application on Microsoft Azure, one of the industry's leading cloud platforms.

Link to access the web application: https://morsehandrehab.eastus.cloudapp.azure.com/

Figure: EagleML Web App to Test data

11. Conclusion

From gathering data to deploying on Azure, every step was taken with care. This project shows how structured data analysis can lead to clear, reliable insights. My methods are designed to offer valuable, data-driven decisions.