Detecting Patterns and Analyzing Outliers in a Drone Dataset

Tech Troopers

June 19, 2024

Abstract

In this report, we present a methodology to detect patterns and analyze outliers in a drone dataset using a regression model. The method leverages the relationship between features to predict a target variable and identifies outliers based on the residuals from the prediction. This approach ensures that new data points that do not fit into the regular pattern are flagged for further investigation. The implementation is detailed, and the results are visualized using scatter plots with outliers highlighted.

1 Introduction

Drones are increasingly used in various fields, from aerial photography to agriculture and delivery services. As the use of drones expands, analyzing drone data becomes crucial for improving performance, safety, and reliability. This report focuses on detecting patterns and analyzing outliers in a large, unlabeled drone dataset. Outliers can indicate anomalies, errors, or novel patterns that warrant further investigation.

We employ a regression model to predict a target variable using selected features from the dataset. By analyzing the residuals (differences between actual and predicted values), we can identify data points that significantly deviate from the norm, marking them as outliers. This method not only helps in understanding the underlying patterns in the data but also in flagging unusual data points that may require attention.

2 Data Analysis

In this section, we perform a comprehensive analysis of the drone dataset to understand its distribution, correlations, and redundancy among features.

2.1 Distribution Analysis

To examine the distribution of key features in the dataset, we create histograms and hexbin plots. These visualizations help identify patterns, skewness, and potential anomalies in the data.

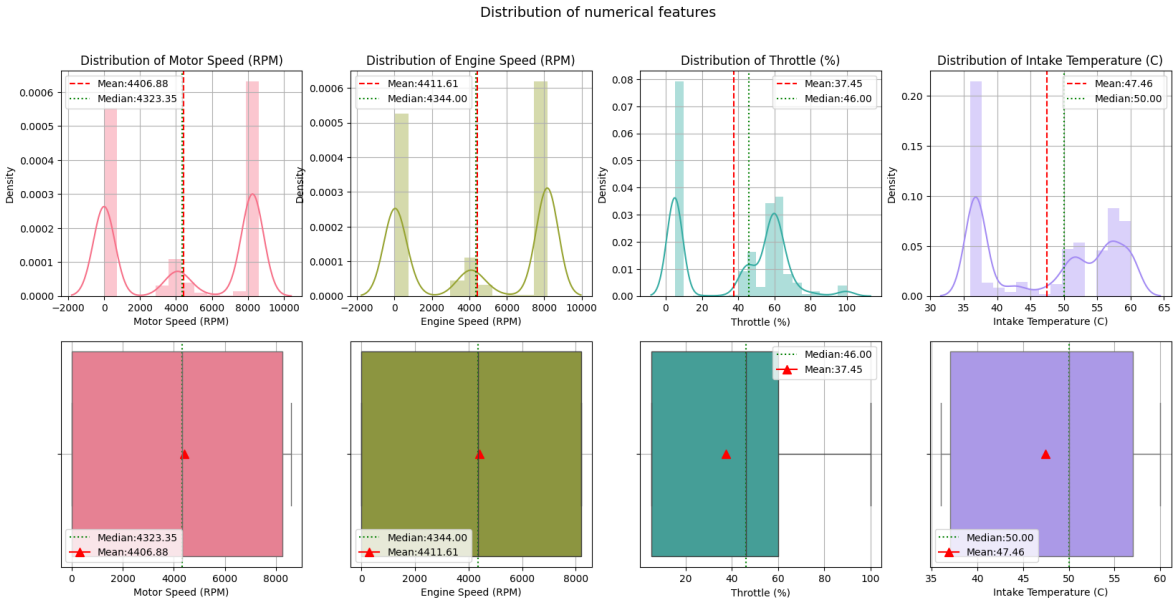

Figure 1: Distribution Analysis 1

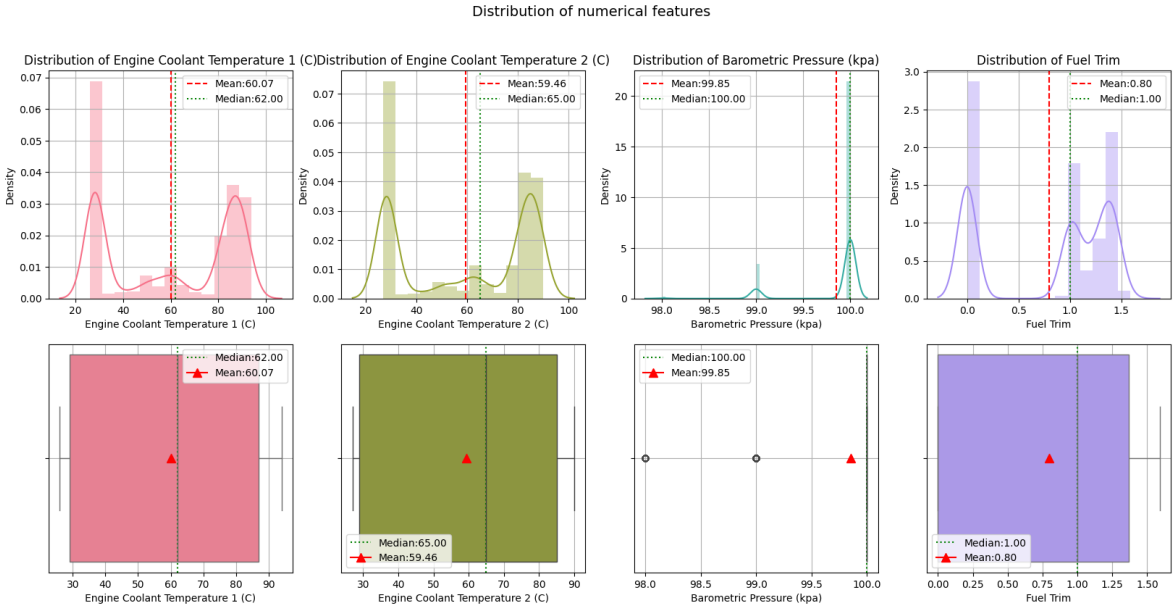

Figure 2: Distribution Analysis 2

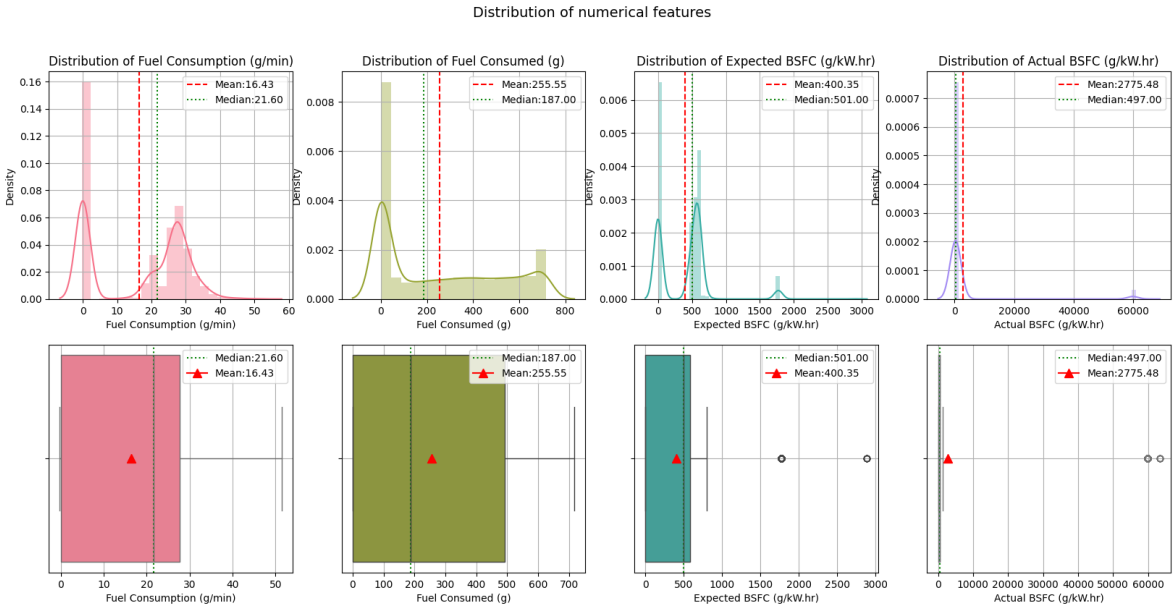

Figure 3: Distribution Analysis 3

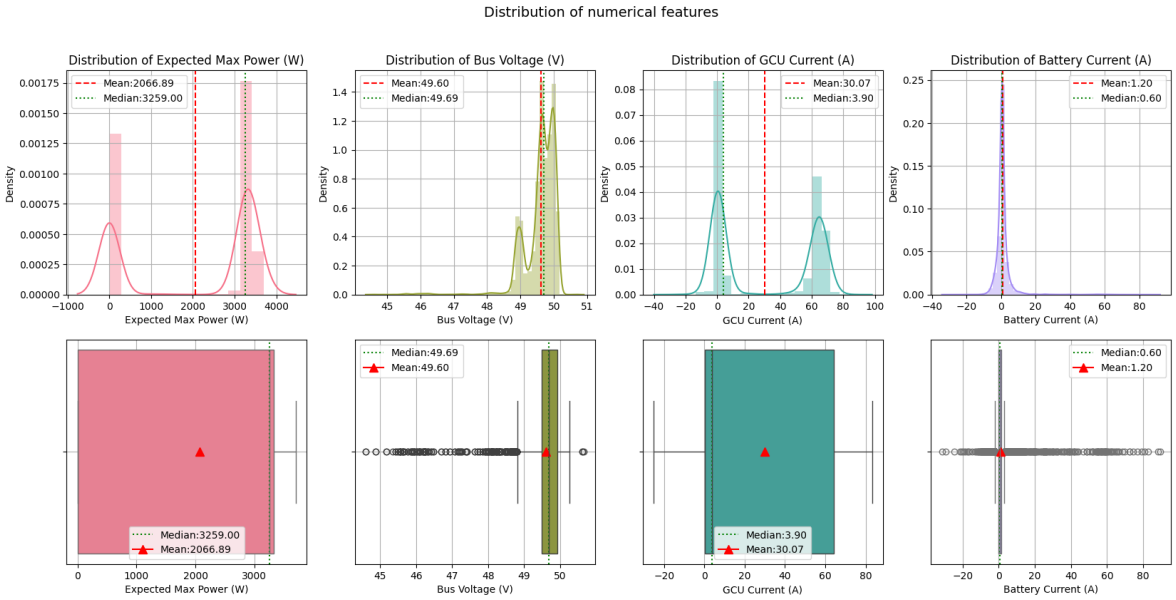

Figure 4: Distribution Analysis 4

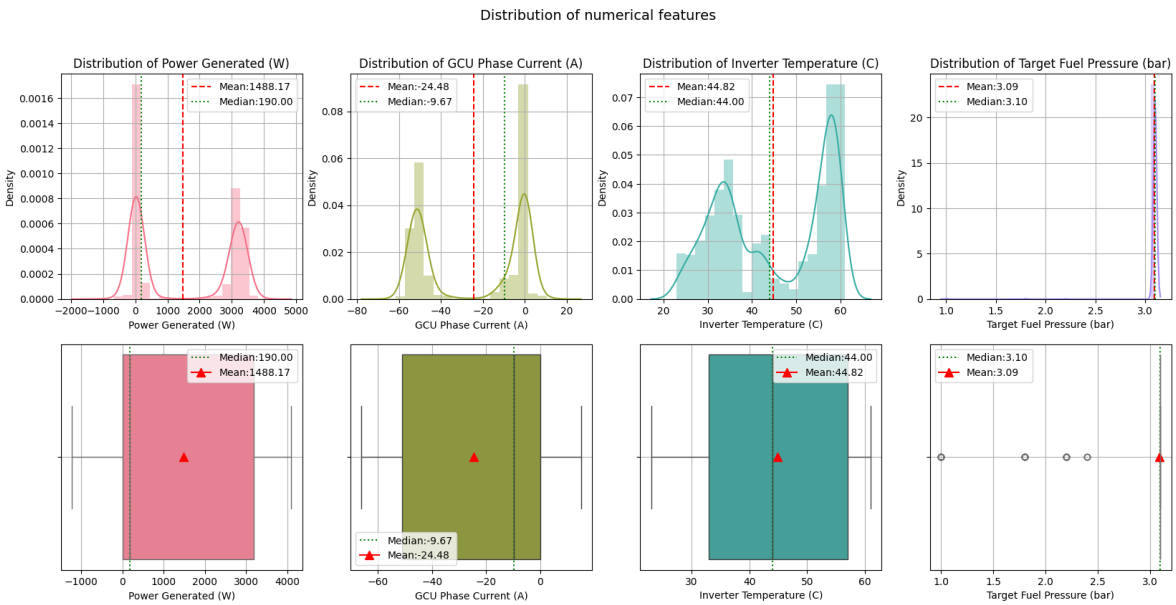

Figure 5: Distribution Analysis 5

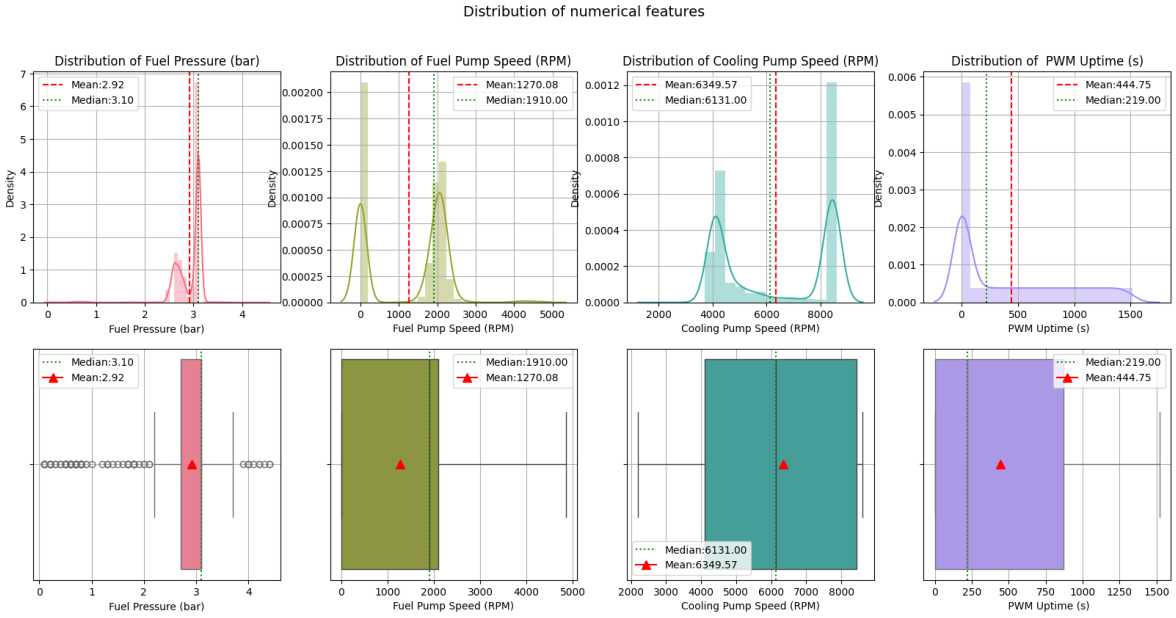

Figure 6: Distribution Analysis 6

2.2 Correlation Analysis

Correlation analysis is performed to identify relationships between different features. A heatmap is used to visualize the correlation matrix, highlighting the strength and direction of the relationships between features.

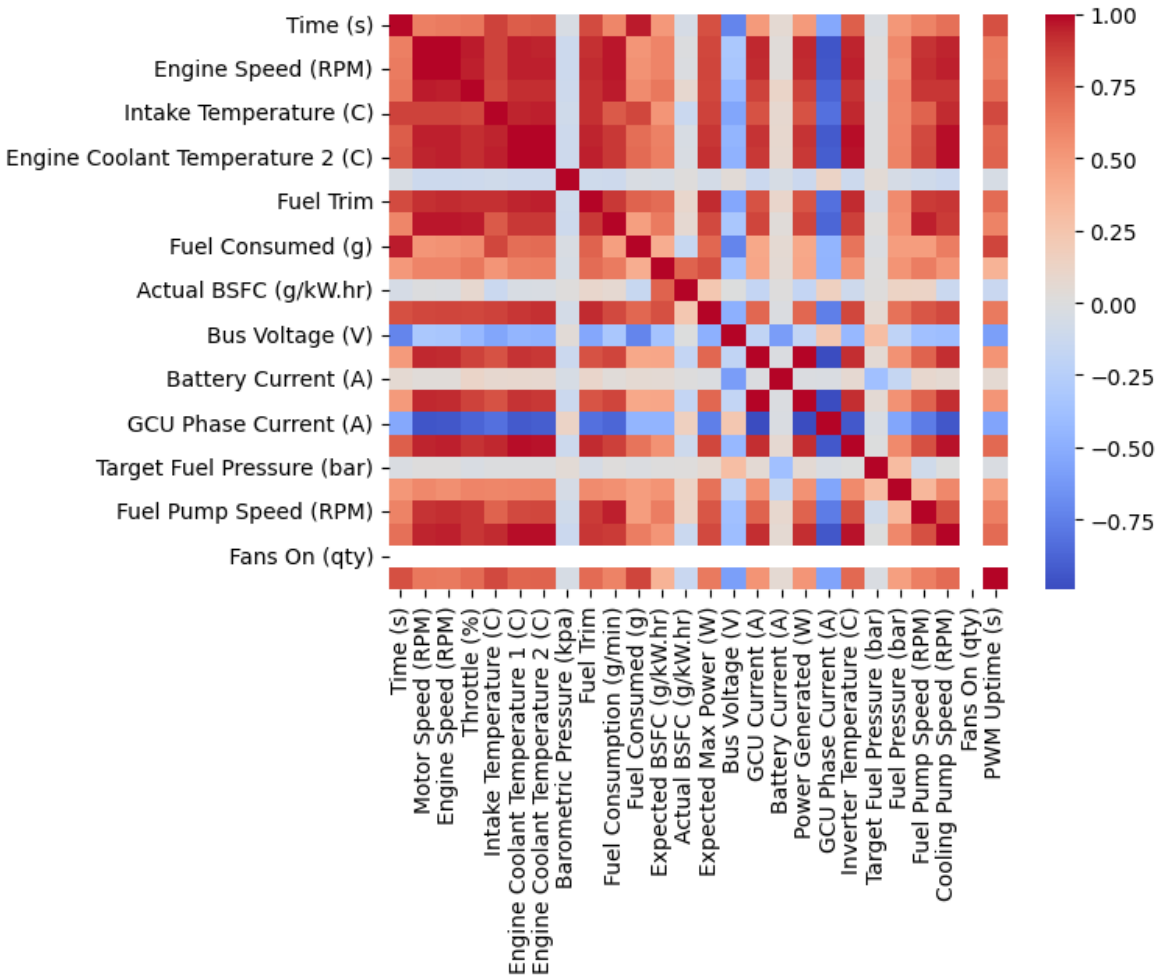

Figure 7: Correlation Heatmap

2.3 Feature Reduction

Feature reduction is a pivotal step in data preprocessing aimed at enhancing model performance and streamlining the dataset for more efficient modeling. By identifying and merging similar features, redundancy is mitigated, leading to a more concise and informative feature space.

One common approach to feature reduction involves comparing the distributions of pairs of features. This comparison is facilitated through the utilization of histograms and hexbin plots, which provide visual insights into the relationship between different features. Additionally, statistical tests such as the t-test are employed to rigorously assess the similarity between feature pairs.

The t-test plays a crucial role in feature reduction by quantifying the statistical significance of differences between the distributions of two features. With an alpha value typically set to 0.05, the t-test evaluates whether the observed differences are statistically significant at a 95

Conversely, if the p-value exceeds the alpha threshold, it suggests that the observed differences between the distributions are not statistically significant. In such cases, the features may be considered redundant, as their distributions exhibit no meaningful deviation from each other. To streamline the dataset and avoid redundancy, one approach is to replace these redundant features with their mean value, effectively consolidating the information they contain.

Feature reduction, guided by statistical tests such as the t-test, contributes significantly to the effectiveness and interpretability of machine learning models. By simplifying the feature space while retaining essential information, feature reduction facilitates more robust model performance and aids in uncovering meaningful insights from the data.

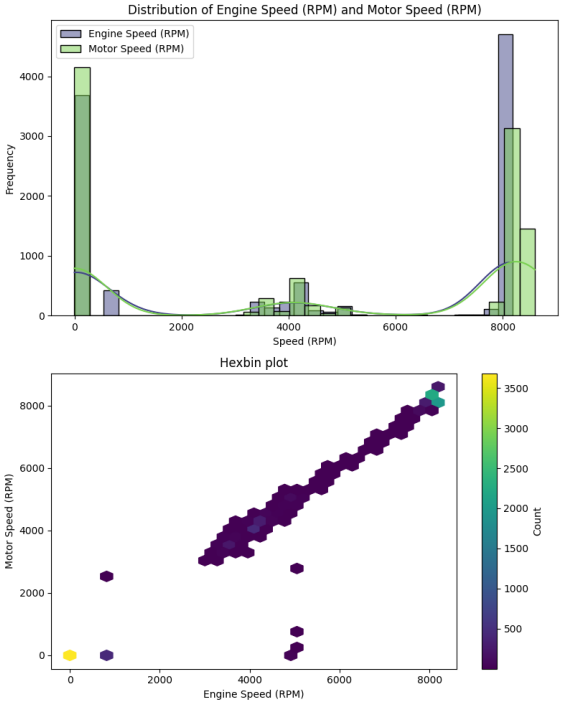

Figure 8: Histograms and hexbins comparing the distributions of two features with a p-value of 0.927. A new feature is formed by taking the average and dropping the previous two features.

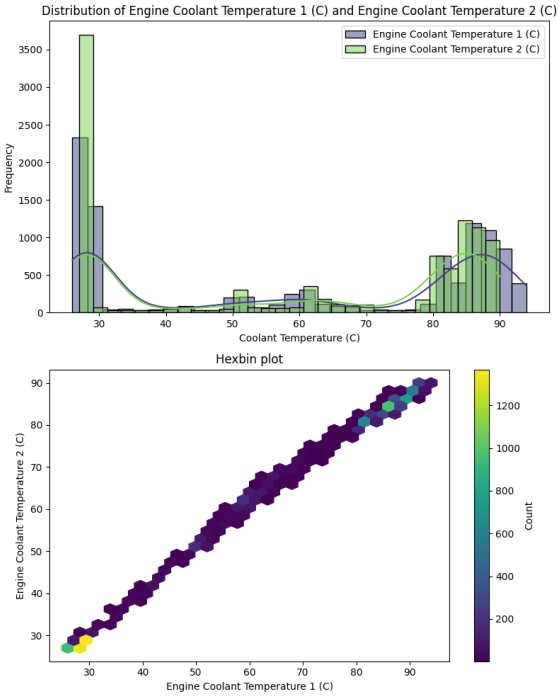

Figure 9: Histograms and hexbins comparing the distributions of two features with a p-value of 0.089. A new feature is formed by taking the average and dropping the previous two features.

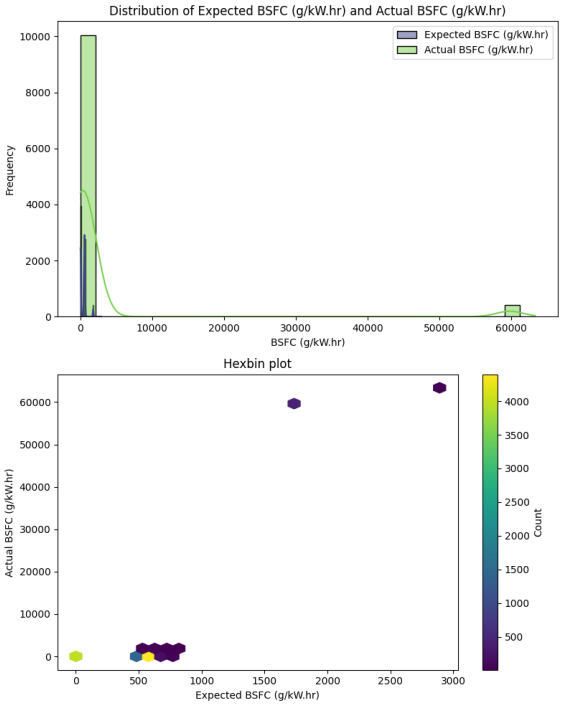

Figure 10: Histograms and hexbins comparing the distributions of two features with a p-value of 5.48×10^{-93}. The features cannot be dropped.

3 Methodology

3.1 Regression Model Approach

The regression model approach involves the following steps:

1. **Extracting Features and Target Variable:** We select specific columns as features (X_names) and a target variable (y_name) from the DataFrame (df).
2. **Creating and Fitting the Models:** We use three different regression models to fit the selected features to the target variable:
   - Linear Regression
   - XGBoost Regressor
   - Random Forest Regressor
3. **Predicting and Calculating Residuals:** Each model predicts the target variable, and residuals are calculated as the difference between the actual and predicted values.
4. **Calculating Standard Deviation of Residuals:** The standard deviation of the residuals is computed for each model to determine the threshold for outlier detection.
5. **Identifying Outliers:** Outliers are identified as data points where the absolute value of the residual is greater than three times the standard deviation of the residuals for each model.
6. **Finding Common Outliers:** The final outliers are determined by finding the common data points identified as outliers by all three models.
7. **Plotting and Highlighting Outliers:** For each feature, a scatter plot is generated with normal data points and outliers highlighted.
8. **Returning Outliers:** The common outliers are returned in a DataFrame.

3.1.1 Mathematical Formulation

Given:

- X : Matrix of feature values
- y : Vector of target variable values
- $y_{\text{pred,model}}$: Predicted values of the target variable by each model
- $\text{residuals}_{\text{model}}$: Differences between actual and predicted values by each model
- σ_{model} : Standard deviation of the residuals for each model

The residuals for each model are calculated as:

$$\text{residuals}_{\text{model}} = y_{\text{pred,model}} - y$$

The standard deviation of the residuals for each model is:

$$\sigma_{\text{model}} = \text{std}(\text{residuals}_{\text{model}})$$

Outliers for each model are identified using the threshold:

$$\text{outliers}_{\text{model}} = \{\text{data point } i \mid |\text{residuals}_{\text{model},i}| > 3\sigma_{\text{model}}\}$$

The final outliers are the common data points identified as outliers by all three models:

$$\text{outliers} = \text{outliers}_{\text{Linear}} \cap \text{outliers}_{\text{XGBoost}} \cap \text{outliers}_{\text{RandomForest}}$$

3.1.2 Implementation

The following pseudo-code outlines the implementation of the methodology:

1. **Define Function:**
   - *Inputs:* Feature columns (*X_names*), target column (*y_name*), DataFrame (*df*), optional model.
   - *Outputs:* DataFrame of outliers.
2. **Extract Features and Target:**
   - Extract values of features and target from DataFrame.
3. **Create and Fit Model:**
   - If no model is provided, use a linear regression model.
   - Fit model with features and target.
4. **Predict and Calculate Residuals:**
   - Predict target values using the model.
   - Calculate residuals as the difference between actual and predicted values.
5. **Calculate Standard Deviation of Residuals:**
   - Compute the standard deviation of the residuals.
6. **Identify Outliers:**
   - Determine threshold as three times the standard deviation of the residuals.
   - Identify data points with residuals greater than the threshold.
7. **Plot and Highlight Outliers:**
   - For each feature, create a scatter plot of feature vs. target.
   - Highlight normal data points and outliers in the plot.
8. **Return Outliers:**
   - Return the DataFrame containing the outliers.
9. **Repeat with different models**
   - We return the process of finding outliers with different algorithms e.g. Linear Regression, Random Forest Regression and XG-Boost Regressor.
   - Only the outliers that are present in all the regression models are considered, while the rest are dropped.

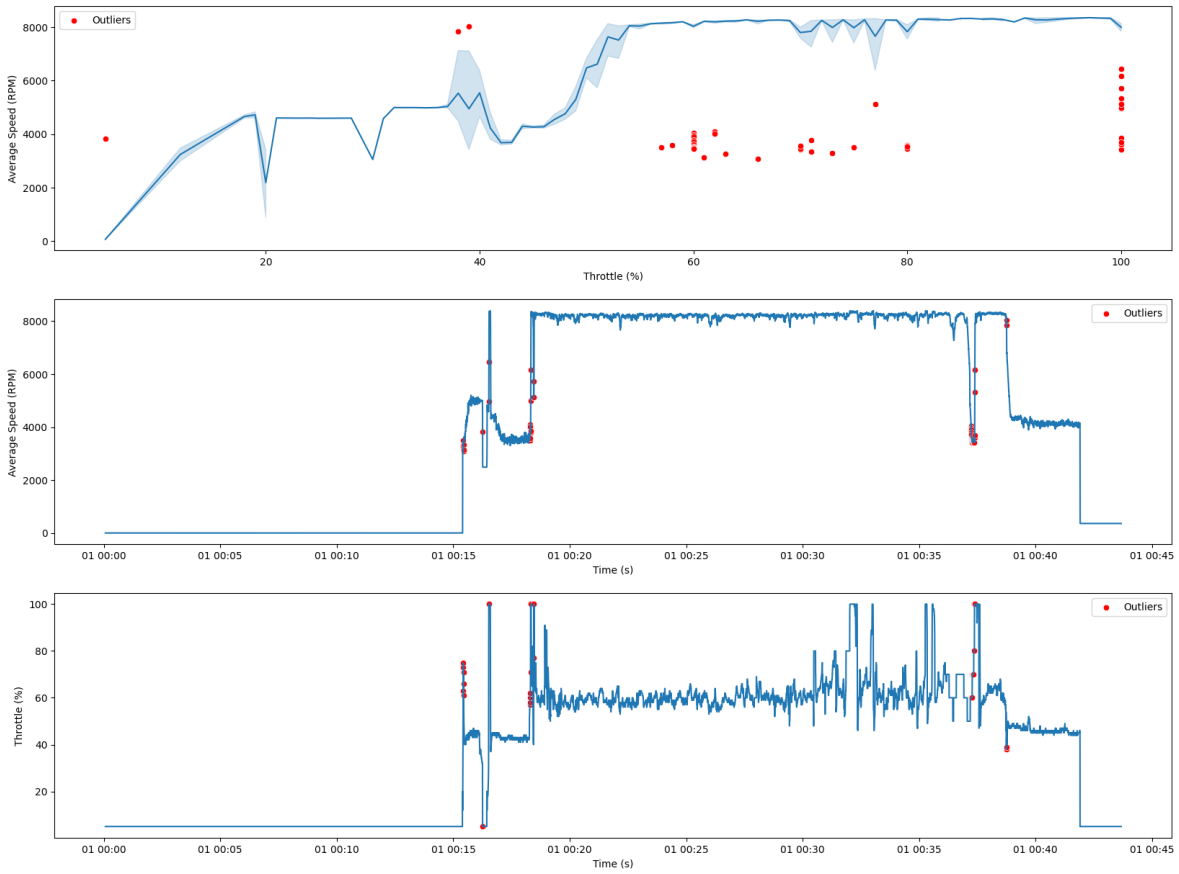

Figure 11: Outliers detected from *Average Speed* as a function of *Throttle (%)*

3.2 Isolation Forest Approach

The Isolation Forest approach involves the following steps:

1. **Extracting Features:** We select specific columns as features (X_names) from the DataFrame (df).
2. **Creating and Fitting the Model:** We use the Isolation Forest algorithm to fit the selected features. Isolation Forest is an unsupervised learning algorithm specifically designed for anomaly detection.
3. **Calculating Anomaly Scores:** The algorithm calculates anomaly scores for each data point, indicating the degree of abnormality.
4. **Identifying Outliers:** Data points with anomaly scores above a certain threshold are identified as outliers.
5. **Plotting and Highlighting Outliers:** For each feature, a scatter plot is generated with normal data points and outliers highlighted.
6. **Returning Outliers:** The outliers are returned in a DataFrame.

3.2.1 Mathematical Formulation

Isolation Forest operates by constructing multiple binary trees (isolation trees) from the data. The algorithm isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. This process continues recursively, producing a tree structure that partitions the data.

The anomaly score for a data point is defined as:

$$\text{score}(x) = 2^{-\frac{E(h(x))}{c(n)}}$$

where:

- $E(h(x))$ is the average path length of point x in the isolation trees.
- $c(n)$ is the average path length of unsuccessful searches in a Binary Search Tree, defined as $c(n) = 2H(n-1) - \frac{2(n-1)}{n}$, where $H(i)$ is the harmonic number.

Data points with higher scores are considered anomalies.

3.3 KMeans Approach

The KMeans approach involves the following steps:

1. **Extracting Features**: Numerical columns are selected as features (X_names) from the DataFrame (df).
2. **Creating and Fitting the Model**: The KMeans algorithm is used to cluster the selected features.
3. **Determining Optimal Number of Clusters**: The optimal number of clusters (k) is determined by evaluating the within-cluster sum of squares (WCSS) across a range of k values. WCSS measures the compactness of clusters.
4. **Identifying Cluster Assignments**: Data points are assigned to clusters based on their proximity to the cluster centroids.
5. **Plotting and Visualizing Clusters**: For each feature, scatter plots are generated where data points are colored according to their assigned clusters.
6. **Returning Cluster Assignments**: The cluster assignments are returned in a DataFrame.

3.3.1 Mathematical Formulation

The KMeans algorithm partitions the data into k clusters by minimizing the within-cluster sum of squares:

$$\text{WCSS}(k) = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$$

where:

- C_i is the i-th cluster,
- μ_i is the centroid (mean) of cluster C_i.

The algorithm iteratively assigns data points to the nearest centroid and updates the centroids until convergence. Data points are assigned to the cluster with the nearest mean, determined by Euclidean distance.

Visualizing the clusters helps in understanding patterns and groupings within the data, aiding further analysis and decision-making processes.

This structured approach ensures clarity and coherence in applying the KMeans algorithm for clustering analysis on the drone dataset.

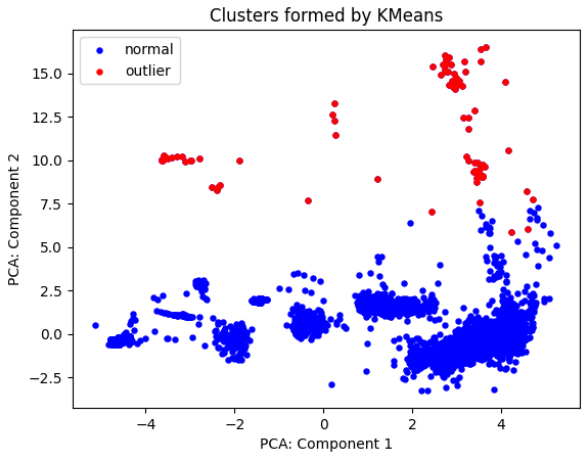

Figure 12: PCA reduced components showing normal and outier points detected by KMeans.

3.4 DBSCAN Approach

The DBSCAN approach involves the following steps:

1. **Extracting Features**: Numerical columns are selected as features (X_names) from the DataFrame (df).
2. **Creating and Fitting the Model**: The DBSCAN algorithm is used to cluster the selected features.
3. **Setting Parameters**: Parameters such as ϵ (eps) and min_samples are set to control the density of clusters. ϵ defines the maximum distance between two samples for them to be considered as in the same neighborhood, and min_samples sets the minimum number of samples in a neighborhood for a data point to be considered as a core point.
4. **Identifying Cluster Assignments**: Data points are assigned to clusters or marked as outliers based on their density connectivity.
5. **Plotting and Visualizing Clusters**: For each feature, scatter plots are generated where data points are colored according to their assigned clusters or marked as outliers.
6. **Returning Cluster Assignments**: The cluster assignments (including outliers) are returned in a DataFrame.

3.4.1 Mathematical Formulation

DBSCAN groups together closely packed points based on two parameters:

- ϵ (eps): Defines the maximum distance between two samples for them to be considered as in the same neighborhood.
- min_samples: Sets the minimum number of samples in a neighborhood for a data point to be considered as a core point.

The algorithm classifies points into three categories:

- Core points: Points that have at least min_samples points within ϵ distance.
- Border points: Points that are within ϵ distance of a core point but do not have enough neighbors to be considered core themselves.
- Noise points (outliers): Points that do not meet the criteria to be core or border points.

Visualizing the clusters helps in understanding the density-based grouping of points and identifying outliers in low-density regions.

This structured approach ensures clarity and coherence in applying the DBSCAN algorithm for clustering analysis on the drone dataset.

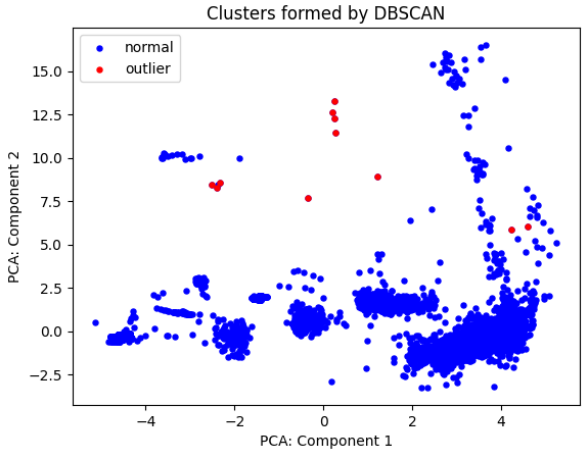

Figure 13: PCA reduced components showing normal and outier points detected by DBSCAN.

3.5 LSTM Model Approach

The LSTM (Long Short-Term Memory) model approach involves the following steps:

1. **Data Scaling**: The dataset is scaled using a 'StandardScaler' to normalize numeric features.
2. **Data Preparation**: Features and target variable are selected from the dataset. Sequences of input-output pairs (X and Y) are created by sliding a window of length T over the scaled data, where T is the number of time steps.
3. **Model Architecture**:
   - Input Layer: Takes sequences of length T with D dimensions.
   - LSTM Layer: Consists of 50 LSTM units to learn temporal dependencies in the sequences.
   - Activation Function: ReLU activation function is applied to introduce non-linearity.
   - Dense Layer: Outputs a single value for regression prediction.
4. **Model Compilation**: The model is compiled with Mean Squared Error (MSE) loss and Adam optimizer with a learning rate of 0.0001.
5. **Training**: The model is trained on X and Y with early stopping implemented to monitor validation loss and restore the best weights.
6. **Validation**: Validation split of 0.5 is used to evaluate model performance during training.

This structured approach ensures that the LSTM model effectively learns from temporal patterns in the data, facilitating accurate prediction of the target variable based on historical sequences.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 20, 1)	0
lstm (LSTM)	(None, 50)	10400
activation (Activation)	(None, 50)	0
dense (Dense)	(None, 1)	51
================================		==========
Total params:		10,451
Trainable params:		10,451
Non-trainable params:		0

Table 1: LSTM model summary

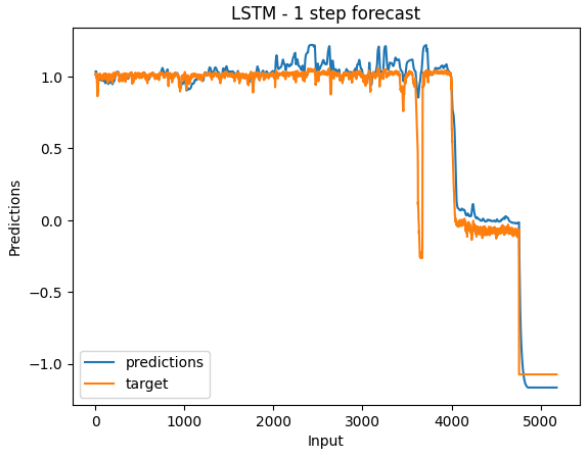

Figure 14: One step prediction for LSTM model

4 Marking Outliers in the DataFrame

We incorporate the outliers obtained from different methodologies in the DataFrame with an 'ALERTalert_index'.

5 Conclusion

This approach effectively identifies and marks outliers in a drone dataset using a regression model. By leveraging the relationship between features, we can predict the target variable and analyze the residuals to detect anomalies. This method ensures that new data points that do not fit the regular pattern are flagged for further investigation.