Objective:

Your task is to write a small Python or R script that predicts the engine rating based on the inspection parameters using only the provided dataset. You need to find all the cases/outliers where the rating has been given incorrectly as compared to current condition of the engine.

This task is designed to test your Python or R ability, your knowledge of Data Science techniques, your ability to find trends, outliers, relative importance of variables with deviation in target variable and your ability to work effectively, efficiently and independently within a commercial setting.

This task is designed as well to test your hyper-tuning abilities or lateral thinking.

Deliverables:

- One Python or R script
- One requirements text file including an exhaustive list of packages and version numbers used in your solution
- Summary of your insights
- List of cases which are outliers/incorrectly rated as high or low and it should be backed with analysis/reasons.
- model object files for reproducibility.

Your solution should at a minimum do the following:

- Load the data into memory
- Prepare the data for modelling
- EDA of the variables
- Build a model on training data
- Test the model on testing data
- Provide some measure of performance
- Outlier analysis and detection

Please answer the following:

1. Briefly describe your approach to this problem and the steps you took

Data Summary:

- Conducted an initial data summary to understand the total number of columns and data points, which provided a baseline overview of the dataset.

Separation of Testing Data:

- Separated the testing data to avoid data leakage, ensuring a time-based split to maintain the integrity of the evaluation.

Feature Segregation:

- Segregated features into categories:
  - **Parent and Child Features**: For example, Engine Oil (Parent) -> Engine_oil_cc_value0 and Engine_oil_cc_value1 (Child).
  - **Other Categorical Features**: Included features like Fuel Type and Comments.
  - **Numerical Features**: Identified and categorized numerical features for further analysis. (Odometer Reading)

Exploratory Data Analysis (EDA):

- Performed EDA on each independent variable to identify important features that influence the target variable.

Feature Derivation:

- Derived new features that had a significant impact on predictions, enhancing the dataset's predictive power. (Ageing)

Time-Based Splitting:

- Conducted a time-based split for training and cross-validation (CV) datasets to ensure temporal integrity.

Data Preprocessing:

- Standardized and applied one-hot encoding to the selected features in the training data to prepare it for modeling.

Model Selection and Hyperparameter Tuning:

- Used Linear Regression and Decision Tree models, followed by hyperparameter tuning to optimize their performance.

Evaluation Metrics:

- Evaluated model performance using metrics such as Mean Squared Error (MSE) and Mean Log Squared Error (MLSE) to assess predictive accuracy.

Error and Prediction Distribution:

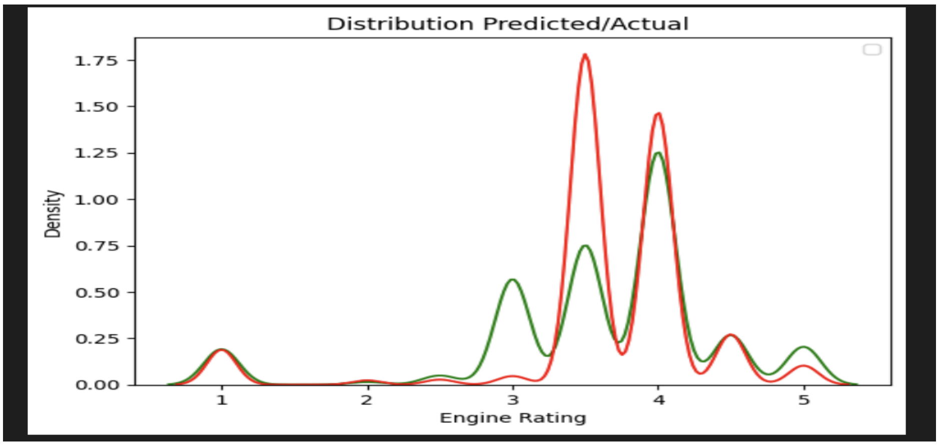
- Analyzed the error and prediction distribution to understand model performance and identify areas for improvement.

2. Basics:

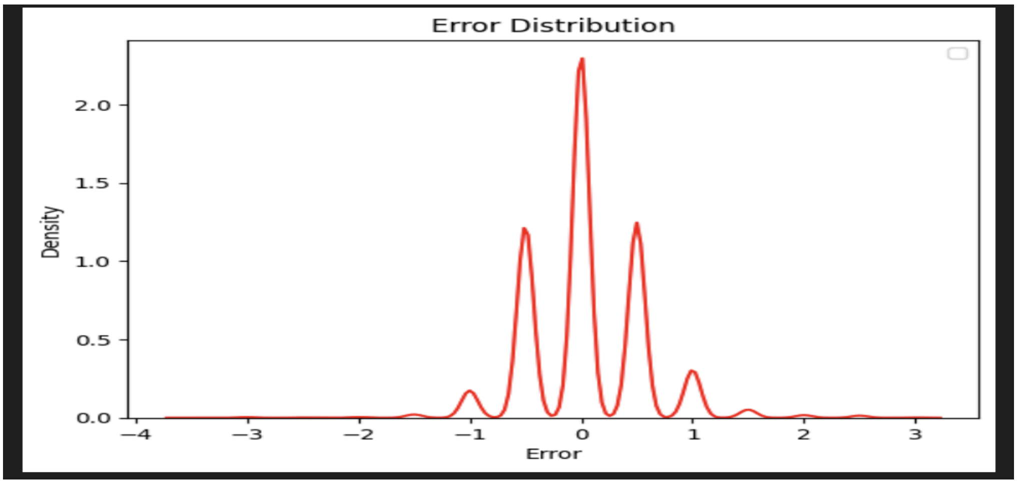
Q1) How well does your model work?

Strengths:

- **Error Distribution**: The error is primarily distributed around 0, showing that the model generally makes accurate predictions.
- **General Performance**: The model seems to handle most ratings well, indicating a good generalization for a majority of the data



- **Base Line model** : Performed significantly well compared to average base line model



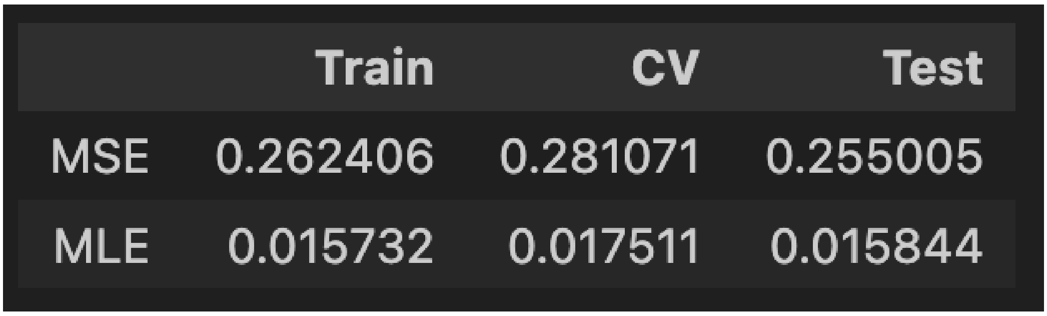
Areas for Improvement:

- **Predicting Rating 3**: The model struggles to accurately predict instances where the rating is 3, leading to misclassifications in this specific range.
- **Error Range**: While most errors lie within a reasonable range, there are still some errors between -1 and 1, indicating potential areas for improvement in model precision, particularly around these edge cases.

Q2)How do you know for sure that's how well it works?

Visual Inspection of Predictions:

- By visualizing the **error distribution** and plotting the predicted vs. actual values, it became clear where the model was performing well and where improvements were needed (e.g., struggling with rating 3). This helped confirm the model's overall decent performance.



	Train	CV	Test
MSE	0.262406	0.281071	0.255005
MLE	0.015732	0.017511	0.015844

Holdout Test Data:

- The model was evaluated on a **holdout test set**, ensuring that the performance metrics weren't inflated due to overfitting. The model's performance on this test set was consistent with the training and validation data, further validating its effectiveness.

Q3) What stats did you use to prove its predictive performance and why?

Mean Log Squared Error (MLSE):

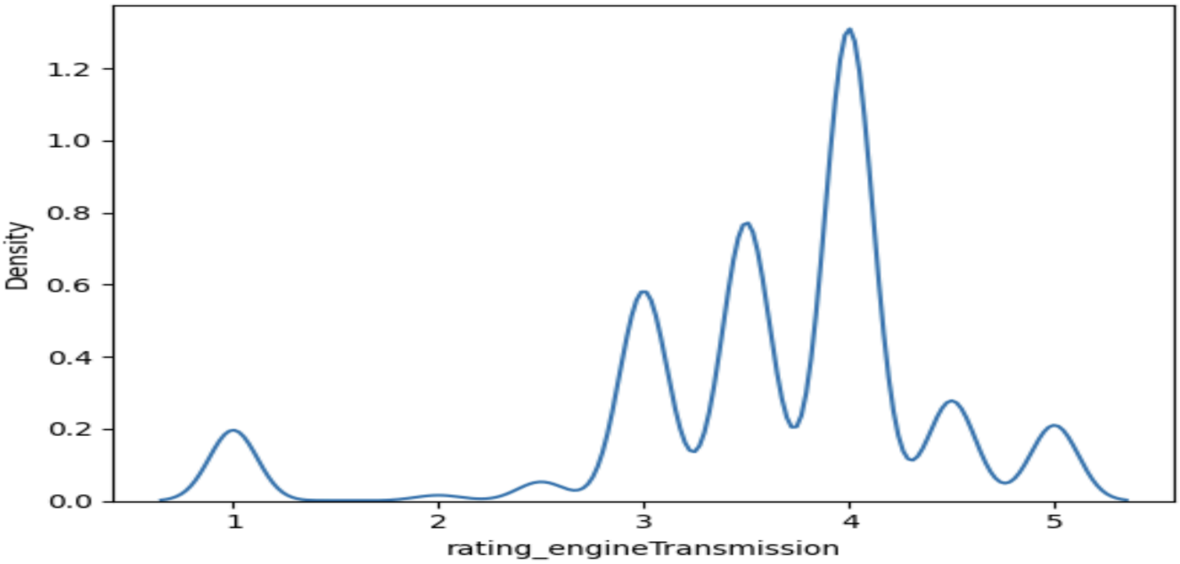
- **Why**: MLSE is typically used to handle cases where the target variable might span several orders of magnitude. It applies a logarithmic transformation, reducing the impact of large errors and focusing more on relative errors, which is useful for data with skewed distributions.
- **Insight**: MLSE is effective when evaluating the model's performance across a wide range of predictions, ensuring that predictions on smaller scales or larger scales are treated proportionally.

- For instance, if the model predicts a 5 instead of a 2, MSLE will penalize this larger gap significantly.

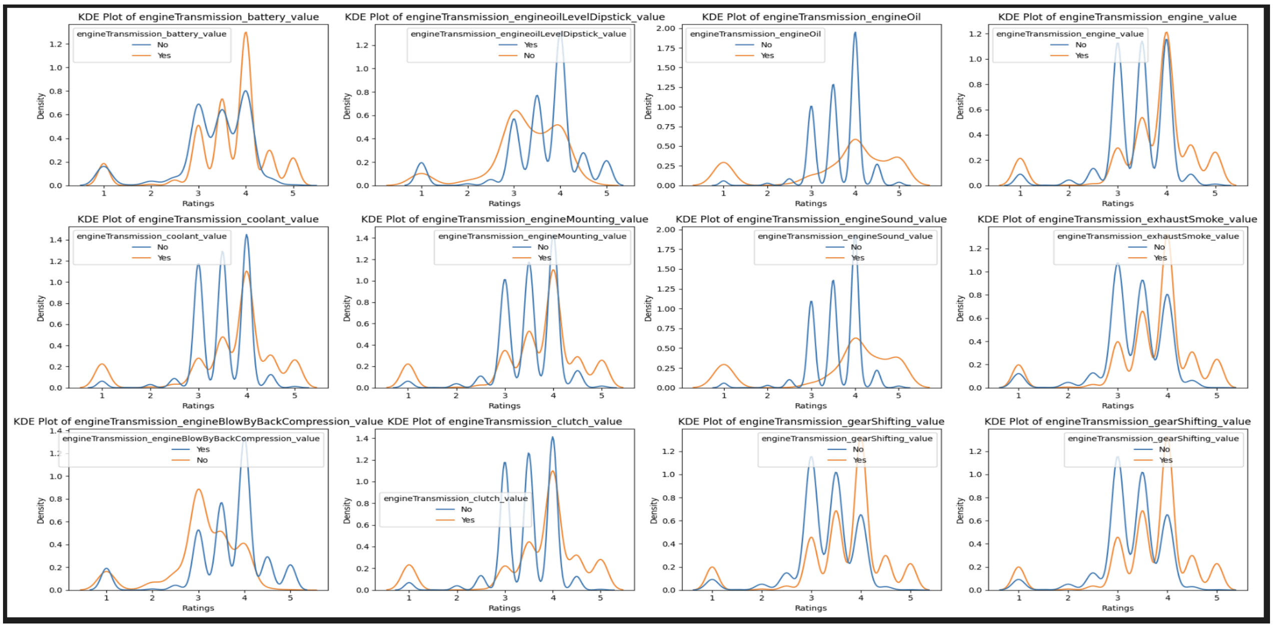
Q4) What issues did you encounter?

- **Handling NULL Values**: At the initial stage, a large number of NULL values in the features posed significant challenges, requiring careful imputation or removal strategies to maintain data integrity.
- **Low Numerical Variables**: The dataset contained very few numerical variables, which limited our ability to detect outliers and assess correlations effectively. This lack of numerical data made it harder to derive meaningful insights.

- **Data Imbalance**: The dataset exhibited a significant imbalance, with a high concentration of ratings between 3 and 4. This skewed distribution allowed even a naive model to achieve a seemingly good score, masking underlying issues with predictive accuracy.



- **Redundant Features**: Many features displayed similar trends, which could lead to multicollinearity. This redundancy complicates model interpretation and may impact model performance by giving undue weight to certain features.



Q5) What insights did you obtain from this data? For example: What features are important? Why? What visualizations help you understand the data?

Year of Registration: This feature showed a strong correlation with the ratings, indicating that newer registrations tend to receive higher ratings.

Ageing (Inspection Time - Registration): This derived feature also demonstrated a clear trend, suggesting that vehicles with longer time since registration are rated lower.

Fuel Type: Different fuel types had varying impacts on ratings, revealing preferences or perceptions related to vehicle performance.

Engine Oil and Engine Sound Condition: These features negatively impacted ratings, indicating that poor engine condition correlates with lower ratings. This was particularly evident in outlier cases.

Visualizations:

- **Normalized KDE Plots**: Kernel Density Estimate (KDE) plots were instrumental in understanding the distribution of features and their relationship with the target variable. They helped identify how the features are spread out and the density of ratings across different values.
- **Violin Plots**: These plots provided insights into the distribution of ratings across different feature categories, showing not only the median but also the spread and density of ratings, which helped in assessing the overall impact of each feature on the target variable.

3. Next steps:

Q1) What other data (if any) would have been useful?

1. Engine power, Brand, Model, Registration Area, Vehicle Pricing etc

Q2) What are some other things you would have done if you had more time?

1. Derive some more insights, extracting valuable features:

2. More data preprocessing

a) Explore more advanced techniques for handling missing data,

3. Experimenting with other models:

a) Ensemble methods, Gradient Boosting models

b) Apply more comprehensive tuning of model hyperparameters using GridSearchCV

By Taking these steps overall goal would be to create a more robust and generalizable model.