CS5228 Final Project Report
Team49: Fuwa Fuwa Time

Yaoyu Cheng[1], Huaxun Chen[2], Wang Sen[3], Senyao Hu[4]
School of Computing, National University of Singapore
Singapore, Singapore
{e1373768, e1351619, e1351109, e1351207}@u.nus.edu

***Abstract*—The aim of this project is to develop a credible and plausible predictive model for used car prices in Singapore. The project uses a dataset from sgcarmart. We first performed an exploratory analysis of the data and found that many features were significantly missing. Subsequently, we employed a number of data preprocessing operations, including estimation, feature engineering, and outlier handling, to improve data quality. We explored step by step and tried different methods for data mining, such as linear regression, tree model, NN and ResNet , and finally chose the ResNet model and took a series of tricks based on it to improve the generalization and stability of the model, and obtained an accurate and stable prediction model, which achieved good results on this dataset.**

***Index Terms*—Regression model, ResNet, Dynamic weight, Clip**

I. Motivation

Singapore is a prosperous, developed country where, in theory, most people have enough financial means to afford a car. However, due to limited land area and high population density, the Singapore government has implemented measures such as the Certificate of Entitlement (COE) to control the number of vehicles. Government intervention, combined with Singapore's high income levels, creates a unique situation in the second-hand car market compared to other countries. For example, the COE price for small to medium cars (Category A) recently reached SGD 102,900 [1], which can be three to five times the vehicle price. For large and luxury cars (Category B), the latest COE price is SGD 113,890 [1]. For luxury vehicles priced at half a million or even more, the COE cost is less significant.

As a result, Singapore's second-hand car market shows a polarized trend. For those with essential transport needs—such as families who need to pick up children or employees commuting to work—buying and selling second-hand cars is often a way to control costs. For these highly circulated small to medium-sized cars, prices tend to be relatively reasonable. On the other hand, luxury car owners, who are generally not financially constrained, are often more concerned with achieving their desired price when selling their cars rather than accepting a market-driven "reasonable" price. This makes luxury car prices particularly difficult to predict.

In the project, data from sgcarmart containing key attributes such as depreciation, arf, omv, and deregistration price are applied for used car selling price prediction. However, the dataset still has quite a lot of missing key data, a large portion of which is inaccurate when filled by direct methods. Thus, finding the correlation between the missing data and other data is crucial and challenging. This is one focus of our work, and the degree of association between data cleaning and feature engineering on the original dataset has a great impact on the accuracy of the predictive model.

Overall, the challenge is how can we propose a model to predict the prices of all kinds of cars to fit different people's needs with similar features input. Our goal is to develop a model that can predict car prices as reasonably as possible by learning from the most important features such as COE, depreciation, and OMV. This model aims to balance predictions between high-priced and mid-to-low-priced vehicles, providing a reference price that maintains a fair balance in the market—neither too high nor too low—thereby protecting the interests of users and promoting a fairer market environment. Our proposed algorithm will be integrated into a second-hand car website as a pricing assistant, offering guidance to both sellers and buyers. This will help customers to make informed decisions in an otherwise complex market.

II. Exploratory Data Analysis

A. Data Overview

We examined the dataset using `describe` and `info`. Key observations:

Numerical Features:
- Features like *mileage* and *depreciation* show high variance, indicating the need for scaling.
- *Indicative_price* has minimal data and may be excluded.

Categorical Features:
- Key features include *make*, *model*, and *fuel_type*.
- *Fuel_type* and *opc_scheme* have many missing values.

Data Quality:
- Columns like *original_reg_date* and *indicative_price* have too many missing entries and will be excluded.
- Missing values in features like *make* and *power* will be addressed via imputation.

B. Missing Values Analysis

A critical part of our EDA involved identifying and analyzing missing values. The key findings are summarized below:

Missing Values Overview: Several important features in the dataset contained missing values. Notably, features such as *depreciation*, *power*, *mileage*, and *fuel_type* had significant numbers of missing entries. The TABLE I below show the details.

TABLE I
MISSING VALUES IN EACH COLUMN (TOTAL COUNT: 25,000)

Feature	Missing Count	Missing Ratio (%)
listing_id	0	0.00
title	0	0.00
make	1316	5.26
model	0	0.00
description	680	2.72
manufactured	7	0.03
original_reg_date	24745	98.98
reg_date	0	0.00
type_of_vehicle	0	0.00
category	0	0.00
transmission	0	0.00
curb_weight	307	1.23
power	2640	10.56
fuel_type	19121	76.48
engine_cap	596	2.38
no_of_owners	18	0.07
depreciation	507	2.03
coe	0	0.00
road_tax	2632	10.53
dereg_value	220	0.88
mileage	5304	21.22
omv	64	0.26
arf	174	0.70
opc_scheme	24838	99.35
lifespan	22671	90.68
eco_category	0	0.00
features	843	3.37
accessories	3813	15.25
indicative_price	25000	100.00
price	0	0.00

Analysis of Missing Values by Brand: A summary of missing values across key features by car brand is as follows:

- **Depreciation**: Luxury brands like *Bentley* and *Rolls-Royce* had higher missing values (12%), while mainstream brands such as *Audi* and *BMW* had fewer missing values.
- **Mileage**: Commercial brands (*Hino*, *Isuzu*) showed over 75% missing, while luxury brands (*Rolls-Royce*, *Ferrari*) had lower missing ratios (below 10%).
- **Power**: Brands like *Hino* and *Isuzu* had 100% missing data for power, while *Audi*, *BMW*, and *Mercedes-Benz* had minimal missing values.
- **Engine Capacity**: Electric vehicle brands (*BYD*, *Tesla*) had 100% missing values, while mainstream brands had minimal missing data.

Text Features with Missing Values: Features like *description*, *features*, and *accessories* had missing values (680, 843, and 3813, respectively), which will be imputed or removed.

Preprocessing Actions:

- **Imputation**: Used group-based, regression, and KNN methods for missing values in *depreciation*, *power*, *mileage*, etc.
- **Dropping Features**: Removed high-missing features like *opc_scheme*.
- **Text Handling**: Replaced missing text with empty strings and transformed via NLP techniques.
- **Brand-Specific Imputation**: Applied medians for brand-specific features like *depreciation*.

This ensures robust, fair predictions across all vehicle types.

C. Outlier Detection and Handling

Outliers can significantly impact model performance, especially in regression problems, by skewing model parameters and reducing predictive accuracy. In our dataset, we applied an outlier detection strategy based on Z-scores, which calculates how far a data point deviates from the mean in terms of standard deviations.

- **Outlier Detection**: We used Z-score analysis to identify outliers in each numerical column of the dataset. Data points with values exceeding 3 standard deviations from the mean were considered as potential outliers. Outliers were identified in multiple numerical features, such as *price*, *mileage*, and *power*.
- **Posssible Handling Strategy**:
  - **Capping**: Replaced extreme values above a certain quantile (e.g., 95%) to reduce anomaly impacts.
  - **Outlier Removal**: Dropped outliers from features where they were deemed unrepresentative (e.g., unusual *mileage* or *depreciation*).
  - **Transformation**: Applied log transformations to skewed features (e.g., *price*, *mileage*) to reduce skewness.
  - **Winsorization**: Replaced top and bottom extremes with values at specific percentiles (e.g., 1%, 99%) to retain all data while minimizing outlier impact.

Our outlier handling strategies aimed to balance between retaining enough data for modeling and ensuring that extreme values do not adversely affect model performance. By capping and transforming data where necessary, and removing unrepresentative outliers, we ensured that the model could learn effectively from representative samples without being disproportionately influenced by anomalous data.

D. Distribution of Target Variable: Price

Exploratory Data Analysis (EDA) focused on understanding the distribution of *price*. We fitted multiple theoretical distributions, including Logistic, Chi-square, T-distribution, Exponential, Weibull, Gamma, Log-normal, Normal, and Johnson SU (see appendix).

1) Key Observations:

- *Price* is **highly right-skewed** with many lower values and a long tail toward higher prices (see Figure 1).
- **Logistic, Normal, Chi-square, and Exponential Distributions** struggled with the data's skewness (7.06) and kurtosis (73.86).
- **T-distribution, Weibull, and Gamma Distributions** moderately captured the peak but missed the long tail.
- **Log-normal and Johnson SU Distributions** fit best, covering the main data but underperforming on extreme tail values.

2) Challenges in Modeling Price Distribution: Key challenges included:

1) **High Skewness and Kurtosis**: Extreme skewness and kurtosis limited fit accuracy across peak and tail.
2) **Long Tail Sparsity**: Few but extreme values in the tail complicated accurate modeling.
3) **Non-uniformity**: A concentration at lower values with occasional high-end prices deviates from typical patterns.

Figure 1 illustrates the skewed nature and long tail of the price data.

3) Conclusion: Given these challenges, no single distribution adequately fits the price data. Future approaches could consider **transforming the target variable** (e.g., log transformation) or using **non-parametric models** and specialized techniques like quantile regression to address the skewness and improve predictive performance.

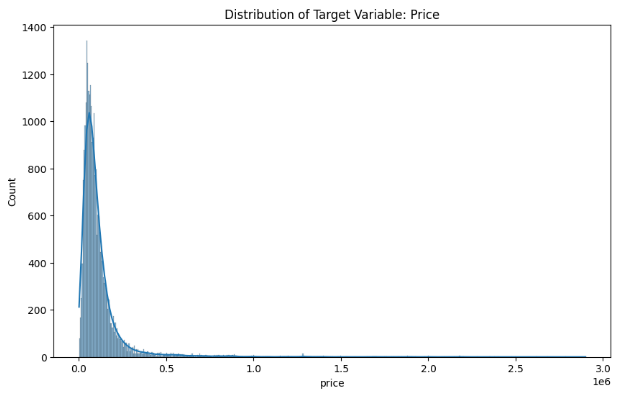


Fig. 1. Histogram of Price with Fitted Distributions: It was observed that none of the tested distributions provided a perfect fit due to high skewness and kurtosis.

E. Overview of Categorical Features

Key categorical features were analyzed to understand their value distributions:

1) Make: The **make** feature represents a range of car manufacturers, from popular brands (e.g., *Mercedes-Benz*, *BMW*) to luxury options (e.g., *Ferrari*).

2) Model: The **model** feature includes a variety of car models, from everyday sedans to luxury sports cars (e.g., *C200*, *Range Rover*, *Aventador*).

3) Title: The **title** feature combines make, model, and details like trim level (e.g., *"Mercedes-Benz C-Class C200 Sport Premium Sunroof"*).

4) Type of Vehicle: Categories under **type of vehicle** include *SUV*, *Luxury Sedan*, and *Sports Car*, representing the dataset's variety.

5) Other Features: Features like **description**, **features**, and **accessories** provide detailed options and upgrades (e.g., premium sound systems).

F. Numerical Features Analysis (Focus on Price)

1) Correlation with Price: The heatmap below shows correlations with *price*. Key features include:

- **Depreciation**: Strong correlation with *price* (0.81), indicating its impact on car value.
- **ARF**: Correlation of 0.89 with *price*, linking higher ARF with higher resale prices.
- **OMV**: Correlation of 0.82, significant for predicting *price*.
- **Deregistration Value**: Highest correlation (0.92) with *price*, making it a key predictor.
- **Power**: Moderate correlation (0.7) with *price*, suggesting engine power boosts resale value.
- **Engine Capacity**: Correlation of 0.44 with *price*, indicating larger engines increase value.
- **COE**: Moderate correlation (0.36) with *price*, showing a limited effect on car value.

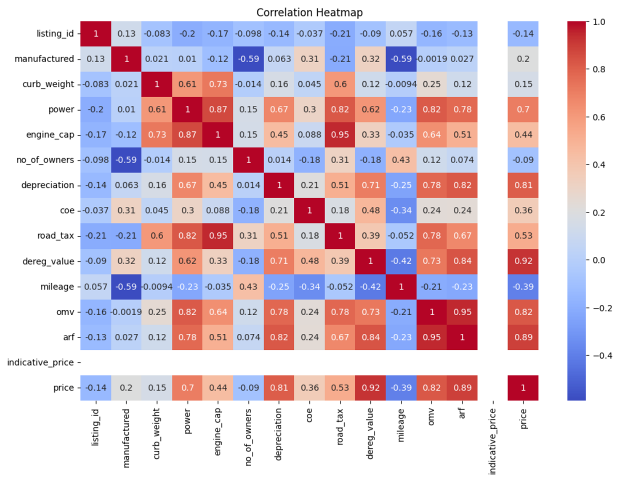


Fig. 2. Correlation Heatmap of Numerical Features with Price

2) Skewness and Kurtosis of Numerical Features: Many numerical features, such as *depreciation* and *ARF*, show high skewness and kurtosis, with most values concentrated at lower levels but some extreme high values.

3) Distribution Analysis: Features like *mileage*, *dereg_value*, *road_tax*, and *depreciation* are right-skewed, indicating a concentration of lower values and a long tail of higher values. Features like *COE* and *engine_cap* exhibit multiple peaks, possibly due to car type or regulatory influences.

4) Key Features for Price Prediction: The top features for predicting *price*, based on correlation, are:

1) **Deregistration Value** (0.92)
2) **ARF** (0.89)
3) **OMV** (0.82)
4) **Depreciation** (0.81)
5) **Power** (0.7)

These will be prioritized in the model to enhance price prediction accuracy.

5) Conclusion and Next Steps: To improve prediction, we'll focus on the top correlated features, apply transformations (e.g., log) to normalize skewed features, and check for multi-collinearity among highly related variables.

G. Categorical Feature Analysis: Make

1) Importance of Categorical Feature for Price Prediction: The *make* feature significantly impacts *price*. We used box-plots and violin plots to explore the relationship between car *make* and *price*.

2) Make Analysis:

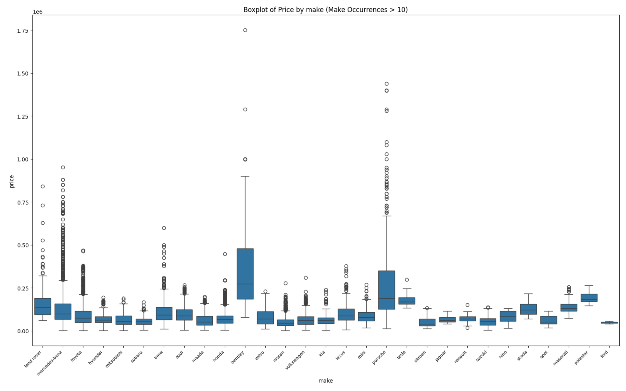


Fig. 3. Boxplot of Price by Make (Occurrences >10)

a) Boxplot of Price by Make: Figure 3 illustrates price distributions across makes with more than 10 instances:

- **Bentley** and **Porsche** have notably higher median prices, reflecting their luxury status.
- Popular brands like **Toyota**, **Honda**, and **Mitsubishi** show lower median prices, aligning with their affordable market segment.
- **Land Rover** and **Tesla** exhibit broader price ranges, covering both premium and performance models.

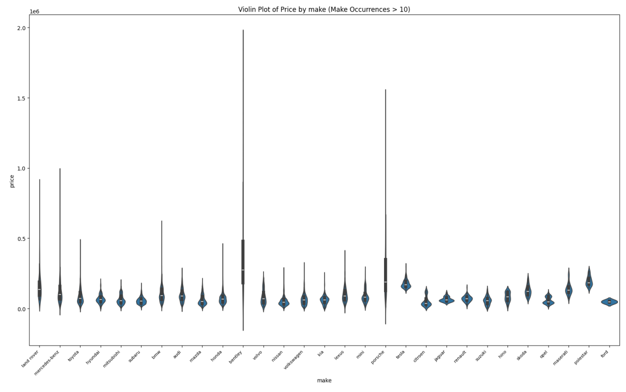


Fig. 4. Violin Plot of Price by Make (Occurrences >10)

b) Violin Plot of Price by Make: In Figure 4, the violin plot reveals:

- Luxury brands (**Bentley**, **Porsche**) show high median prices with long tails, reflecting a wide price range based on model configurations.
- Mainstream brands (**Toyota**, **Honda**) have compact distributions, indicating consistent, lower pricing.
- **Land Rover** and **Tesla** display bimodal distributions, likely due to distinct model types (standard vs. luxury/performance).

3) Conclusion: The *make* feature significantly influences price, with luxury brands having higher and more variable prices than mainstream brands. These insights will inform feature selection in our predictive model.

III. Data Cleaning and Preprocessing Steps

A. Data Loading and Initialization

The datasets were initially loaded using `pandas` in Python. To speed up the loading process, parallel processing was used via `joblib`, allowing both training and testing datasets to be loaded simultaneously. Key fields of interest were identified, and preliminary operations such as data type conversion and missing value counts were performed.

B. Handling Missing Values

Missing values in columns such as `depreciation`, `OMV` (Open Market Value), `ARF` (Additional Registration Fee), and others were addressed using a systematic approach:

1) **Government Algorithm for OMV and ARF**: Missing values in `OMV` and `ARF` were first filled using the government-prescribed algorithm [2] based on official regulations:

$$\text{ARF} = \begin{cases} \text{OMV}, & \text{if OMV} \leq 20,000 \\ 20,000 + 1.4 \times (\text{OMV} - 20,000), & 20,000 < \text{OMV} \leq 40,000 \\ 20,000 + 1.4 \times 20,000+ & \\ \quad 1.9 \times (\text{OMV} - 40,000), & 40,000 < \text{OMV} \leq 60,000 \\ 20,000 + 1.4 \times 20,000+ & \\ \quad 1.9 \times 20,000+ & \\ \quad 2.5 \times (\text{OMV} - 60,000), & 60,000 < \text{OMV} \leq 80,000 \\ 20,000 + 1.4 \times 20,000+ & \\ \quad 1.9 \times 20,000+ & \\ \quad 2.5 \times 20,000+ & \\ \quad 3.2 \times (\text{OMV} - 80,000), & \text{otherwise} \end{cases}$$

2) **Median by Similar Titles**: Missing values in `depreciation`, `OMV`, and `ARF` were filled using the median of similar titles, based on the first three words of the `title` column.
3) **Model Median Imputation**: For any remaining missing values, we grouped by `make` and `model` and filled missing entries with the median within each group.
4) **K-Nearest Neighbors Imputation (KNN)**: For features like `manufactured`, `engine_cap`, and `curb_weight`, missing values were filled using `KNNImputer` based on similar instances.
5) **Global Mean Imputation**: Finally, any remaining missing values were filled with the overall mean of each column.

C. Categorical Adjustment

Vehicles were classified as 'PARF' or 'COE' cars based on the `category` column, with a new `car_classification` column created for further feature engineering. Missing values in `make` were filled with the most frequent value within similar `model` groups.

D. Feature Engineering

1) Remaining Years of COE Calculation: The `remaining_years_of_coe` was computed based on the vehicle registration date and COE extension status. This was used to improve `depreciation` and `calculated_price` estimates.

2) PARF Value Calculation: The PARF value was determined using the vehicle's registration date and either OMV or ARF:

- For vehicles registered after March 1, 2013: parf_price $= 0.5 \times$ ARF
- Between March 1, 2008, and March 1, 2013: parf_price $= \min(0.5 \times \text{OMV}, 0.5 \times \text{ARF})$
- Before March 1, 2008: parf_price $= \min(0.55 \times \text{OMV}, 0.5 \times \text{ARF})$

3) Calculated Price Calculation: To compute an calculated vehicle price, the following formula was used:

$$\text{calculated\_price} = (\text{depreciation} \times \text{remaining\_years\_of\_coe}) + \begin{cases} \text{parf\_price} & \text{if car\_classification = 'parf car'} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where `parf_price` represents the PARF rebate value, which only PARF cars can receive when being deregistered.

4) Price Adjustment: To ensure realistic pricing, `calculated_price` was adjusted within the range defined by `max_price` and `min_price`:

$$\text{calculated\_price} = \begin{cases} \text{max\_price} & \text{if c\_price} > \text{max\_price}, \\ \text{min\_price} & \text{if c\_price} < \text{min\_price}, \\ \text{calculated\_price} & \text{otherwise.} \end{cases}$$

This adjustment ensures prices fall within the observed market range.

5) Text Feature Processing: Text features like `title`, `description`, and `category` were processed using TF-IDF vectorization and SVD for dimensionality reduction. Additionally, sentence embeddings were created using Sentence-BERT and reduced using PCA for modeling.

6) Categorical Encoding: Categorical features, such as `make`, `model`, and `type_of_vehicle`, were encoded using target and frequency encoding to capture their relationship with the target variable `price`.

7) Feature Interactions and Polynomial Features: Interaction terms and polynomial features were created for important variables like `make_encoded`, `model_encoded`, and `depreciation` to capture non-linear relationships.

8) Data Augmentation: Data augmentation was applied to underrepresented classes (e.g., luxury sedans and sports cars) by adding noise to numerical features.

E. Final Adjustments and Results

Bias adjustments were applied to the `calculated_price` using:

$$\text{calculated\_price} = (\text{calculated\_price} - 380) \times 0.943$$

This corrected for discrepancies and aligned the predicted prices with actual market values.

F. Conclusion

The data cleaning and processing steps outlined above produced a refined dataset with minimal missing values, consistent categories, and computed features that enhance the predictive quality of the model. These steps, including handling missing values systematically, feature engineering, text processing, and categorical encoding, have enabled a robust foundation for further analysis and model training. Interaction features, text embeddings, and carefully handled categorical features all contribute to a dataset that is primed for effective price prediction modeling.

IV. Data Mining Methods

A. Overview

In this section, we provide an overview of the data mining methodology we employ. Our data mining is a step-by-step process of exploratory experimentation, where we first used a linear model as a foundation, then a tree model, then an FNN model, and finally an advanced ResNet-based CNN model. This reflects our pursuit of improving accuracy and model robustness.

B. Linear Models

At the very beginning of the project, we prioritized linear regression models, such as `Lasso` and `Ridge`, as the baseline method because they run faster, are more interpretable and more stable than other models, and help to filter the features.`Lasso` regression selects features through L1 regularization, thus automatically screening out redundant variables and laying a streamlined and effective feature foundation for subsequent modeling. `Ridge` regression, on the other hand, effectively suppresses multicollinearity among features through L2 regularization, making it suitable for cases where there is a correlation between the features.

C. Tree-Based Models

In the early stages of the project, we have used linear regression models for exploration, but as we gained a better understanding of the problem, we decided to introduce a variety of tree-based models to further improve the accuracy of the predictions and the performance of the models. In used car price prediction, there are nonlinear relationships between many factors (e.g., make, model etc.), and tree-based models are able to automatically capture these complex patterns.

Random Forest is an ensemble learning method that combines several decision trees, which together manage to minimize overfitting and thus increase the stability and accuracy of the model. XGboost and LightGBM are two gradient boosting models which achieve the fastest training speed and high efficiency on large scale data. GradientBoosting Regressor, on the other hand, optimizes and corrects errors in a step-wise fashion; therefore capturing complex relationships in the data more efficiently. In contrast, CatBoost natively uses categorical features which improves model performance and removes the burden of extensive preprocessing.

D. Neural Network (NN) Model

1) Motivation for Neural Networks: Given the limitations of the tree-based models in learning non-linear and complex relationships, we shifted our approach to a **fully connected Neural Network (NN)**. Neural networks can model complex functions, making them well-suited for tasks with intricate dependencies among features. Our NN model was structured with **five fully connected layers**, leveraging both depth and feature engineering to improve predictive performance.

2) Model Architecture and Training: The NN model consisted of five dense layers with 512, 256, 128, 64, and 1 unit(s), respectively. Each hidden layer used `ReLU` activation, while the output layer used a `linear` activation function for regression. To prevent overfitting, we used `L2 regularization` with a weight of 0.02. The optimizer was `AdamW`, and the loss function was `Mean Squared Error (MSE)`, with `Root Mean Squared Error (RMSE)` as the evaluation metric.

3) Results and Analysis: The NN model was trained using **5-fold Cross-Validation** with a batch size of 512 and up to 1000 epochs. **EarlyStopping** and `ReduceLROnPlateau` were applied to prevent overfitting and adjust the learning rate dynamically. The model achieved an average RMSE of around 16914 on the test set, which was an improvement over the tree-based models. However, it became evident that the NN struggled with learning higher-order feature interactions, motivating us to explore more advanced architectures.

E. ResNet Model

1) Motivation for ResNet: The performance of the NN model highlighted the need for an architecture that could capture subtle feature interactions more efficiently. We therefore choose **ResNet** architecture, which overcomes the problem of vanishing gradients and also learns deep residual mappings to improve accuracy. We utilize **skip connections** to allow the model to learn both raw inputs and residuals, thus improving learning ability and stability.

2) Model Architecture and Training: The ResNet model was designed with the following structure:A `Conv1D` layer with 64 filters, followed by `BatchNormalization` to stabilize training.And two ResNet blocks, each consisting of two convolutional layers with skip connections, allowing the model to learn residual mappings efficiently. In addition, they consist of global pooling and dense layers, regularization and optimizer.

3) Results and Improvements: The ResNet model demonstrated a significant improvement in validation RMSE, achieving an average of around 12,000 across folds. This improvement can be attributed to ResNet's ability to model more complex interactions between features through its skip connections, effectively capturing high-order relationships that were previously missed by both the tree-based models and the standard NN.

F. Comparison and Justification of Design Choices

1) Linear Model vs. Tree-Based Models: We start with an initial exploration using linear models that benefit from pre-existing guidance on formula computation. This guidance helps linear models capture the linear relationship between prices and features more accurately, thus outperforming tree-based models in some contexts. Although tree-based models are better able to handle nonlinear relationships in the data and capture complex interactions between features, they can be affected by some nonlinearities and do not perform as well as models with explicit linear relationships.

2) Tree-Based Models vs. Neural Networks: The tree-based models, while effective for an initial analysis, could not fully capture the intricate, non-linear relationships present in the dataset, particularly for high-priced vehicles. Their interpretability and ability to handle categorical variables were beneficial; however, their generalization capability was limited compared to neural networks. The NN model showed improvements in capturing non-linear dependencies, but it lacked the deep learning capacity that was ultimately needed.

3) Neural Network vs. ResNet: The NN model laid the foundation for understanding the relationships between features and car prices. However, the ResNet model significantly outperformed it by addressing the limitations of deeper networks through residual learning. The skip connections in ResNet effectively mitigated the vanishing gradient problem, allowing the model to learn deeper and more nuanced representations of the input features. This was especially important in capturing interactions between variables like `depreciation`, `car_age`, and `power`.

G. Final Model Selection

Based on the results from cross-validation and performance comparisons, the **ResNet** model was selected for final training and prediction. Based on the ResNet model, we further select different features to suit different price ranges, and ultimately combine these predictions.

H. Conclusion

The data mining process evolved from using **tree-based models** to **fully connected Neural Networks**, and finally to a **ResNet-based model**. Each step in this progression addressed specific limitations observed in the preceding model. Tree-based models provided a solid baseline, while the NN model improved upon it by capturing non-linear relationships. The ResNet model ultimately achieved the best performance by leveraging residual learning, demonstrating the effectiveness of deep learning techniques for this task. The progression of models and careful consideration of each design decision ensured a well-justified and methodologically sound approach to predicting car resale prices.

V. Evaluation & Finetune

A. Performances

See TABLE II

TABLE II
RMSE Performance of Different Models on the Test Set

Model	RMSE
Weighted model with clip	**10,959.18**
Weighted model	10,960.13
ResNet model with clip	14,822.93
ResNet model	14,827.82
Neural Network model with GBM	16,914.39
Gradient Boosting Regressor	17,559.01
Lasso regression	17,837.42
Random Forest	17,987.29
Linear regression	18,099.86
CatBoost	20,421.33
LightGBM	25,055.07
XGBoost	25,540.79

B. Analysis

1) Linear Models: The results on the test set show that the Lasso regression and Ridge regression models exhibit robust performance, with an RMSE of 17,837 for Lasso regression and 18,099 for Ridge regression. When compared with the results of the tree model, the results of the linear model are surprising. This may be due to the fact that sgcarmart's formula for car prices is itself linear, and this formula uses features such as depreciation, omv, and arf, and our refined data cleaning allows this missing data to be filled with a small error, which makes the linear model perform very well.

2) Tree-Based Models: XGBoost and LightGBM failed to outperform linear models as well as random forests in this project. The possible reason for this is that XGBoost and LightGBM failed to take full advantage of their strengths in nonlinear modeling due to the strong linear relationship between some of the features in used car price prediction. After tuning, GradientBoosting Regressor shows the best prediction performance. Therefore, despite the strong theoretical advantages of XGBoost and LightGBM, in this project, GradientBoosting Regressor ultimately achieves the best results on the test set through efficient error correction.

3) Neural Network Models: The Neural Network (NN) model performed well, achieving an RMSE of 16,914.39 on the test set, primarily due to the following reasons: first up is Complex Relationship Modeling, NN model can effectively learn and model these complex dependencies between features such as mileage, power, and depreciation. Secondly,it may be Feature Engineering Contributions, Before training the NN model, we used a large amount of feature engineering to greatly improve its performance. The third is Regularization and Optimization Strategies, the NN model incorporated `L2 regularization` in each layer.The `EarlyStopping` and `ReduceLROnPlateau` callbacks further ensured that: the training process stopped when no significant improvement was observed, dynamically adjusted the learning rate when necessary.

4) ResNet Model: The ResNet model achieved an even better performance, with an RMSE of 14,827.82 on the test set. The primary reasons for the superior performance of ResNet over the standard NN model are as follows: First up is Skip Connections and Residual Learning, they make the optimization process more efficient and stable. And then is Convolutional Feature Extraction, they can effectively capture local patterns and relationships among the input features. Following is Enhanced Generalization with Batch Normalization, This helped the model generalize better to unseen data and led to better validation performance compared to the standard NN model.

5) Feature Selection: In order to determine the most influential features for predicting car resale prices, we utilized the SHAP (SHapley Additive exPlanations) analysis to select and prioritize the most important features with reference to the SHAP values.

Figure 5 presents a summary plot of SHAP values, which visualizes the distribution of feature contributions across different samples. From this plot, we can observe features such as `car_age` and `depreciation` show a high SHAP value and it indicates that they are positively correlated with price.

Moreover, we could select different combinations of features based on their SHAP values. For example, the `make_encoded` and `model_encoded` features may improve high-price car predictions but decrease accuracy for low-price cars. We used this characteristic to train specific models for different price ranges and then combined them to create a balanced model.

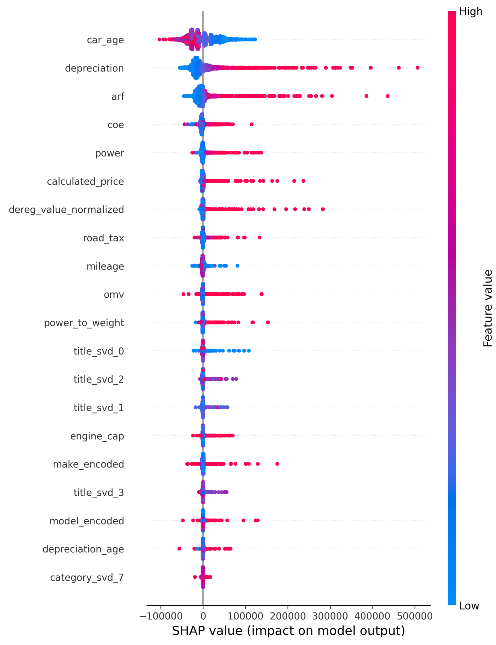


Fig. 5. SHAP Summary Plot of Feature Contributions

In conclusion, the SHAP analysis played a crucial role in guiding our feature selection process, allowing us to identify the features with the greatest predictive power and thereby improve the overall model performance.

6) Dynamic Weight: Dynamic weighting adjusts prediction weights based on the value range to improve accuracy. By combining the predictions from different models trained on selected features, we utilized a dynamic weighting strategy to optimize the final model's performance. The goal was to leverage the strengths of each model, such as one model excelling at predicting high-priced cars while another was better for low to mid-priced cars, and create a well-balanced ensemble that minimized RMSE across different price segments.

Initially, we selected two models whose RMSEs were similar on the training set but exhibited larger differences when comparing their prediction errors across different price ranges. This approach allowed us to identify complementary models for combination. Using dynamic weighting, we iteratively combined these models to achieve a balanced final model, and then repeated the process with this newly combined model against others until optimal weight factors were determined.

Here is how dynamic weight works:

Define a range of quantiles to perform a grid search over possible boundaries:

- For each combination of lower quantile q_{low} and upper quantile q_{high} such that $q_{\text{low}} < q_{\text{high}}$:
  1) Calculate price thresholds low_threshold and high_threshold based on q_{low} and q_{high}.
  2) Divide the training data into three ranges:
     - Low price range: `price` $<$ low_threshold
     - Medium price range: `low_threshold` $\leq$ `price` $\leq$ high_threshold
     - High price range: `price` $>$ high_threshold
  3) For each range, find the weight w that minimizes the RMSE between the combined predictions of two models and the true values:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - (w \cdot \hat{y}_{1,i} + (1-w) \cdot \hat{y}_{2,i})\right)^2}$$

  4) Combine the models dynamically by applying the best weight for each range and calculate the overall RMSE for the current quantile configuration.
  5) Update the best boundaries and weights if this RMSE is lower than the previous minimum.

a) Training Data Prediction Diff Before Dynamic Weight: **Root Mean Squared Error (RMSE)**: 13718.34

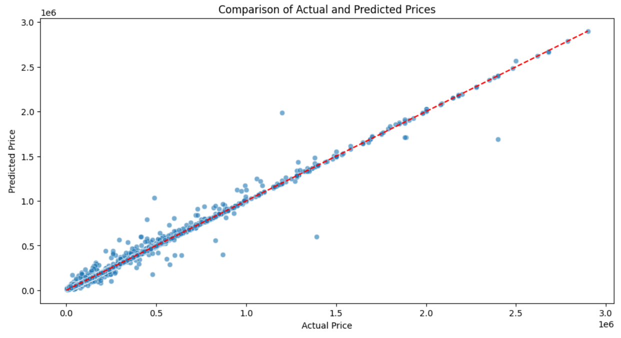


Fig. 6. Training Data Prediction Diff Before Dynamic Weight

b) Training Data Prediction Diff After Dynamic Weight: **Root Mean Squared Error (RMSE)**: 6374.24

The application of dynamic weighting significantly improved the model's performance, as evidenced by the reduction in both the Root Mean Squared Error (RMSE). Specifically,

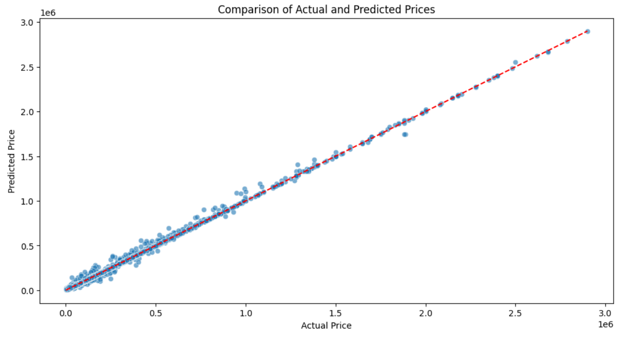


Fig. 7. Training Data Prediction Diff After Dynamic Weight

the RMSE dropped from 13718.34 to 6374.24. These improvements indicate that dynamic weight optimization, by adjusting prediction weights according to different price ranges, helped the model more accurately capture the nuances of the data. This technique effectively reduced the prediction error, particularly for outliers or data points in varying price ranges, suggesting that dynamic weight adaptation allows for a more tailored and precise model performance.

7) Clip: In order to ensure that the predicted car prices are within a reasonable range, we applied a method called **clipping**.

To implement this, we trained two **linear regression models** to predict the upper and lower bounds for each car's price.

The ResNet model was used to predict the car prices, and then these predictions were passed through the clipping process. Using `np.clip()`, the predicted price for each vehicle was restricted to lie within the range given by the minimum and maximum price models:

- If the predicted price was below the minimum predicted value, it was set to the minimum value.
- If it was above the maximum predicted value, it was set to the maximum value.
- Otherwise, the predicted price was kept as is.

This approach helped in improving the stability and reliability of the model's predictions, especially for high-priced vehicles where predictions can be more prone to errors. By applying clipping, we ensured that the final predicted car prices remained within a realistic and practical range, thereby enhancing the credibility of the results.

References

[1] Sgcarmart, "COE Prices," Sgcarmart. [Online]. Available: https://www.sgcarmart.com/coe-price. [Accessed: Nov. 14, 2024].

[2] Sgcarmart, "Understanding OMV, PARF, and COE Rebates for Your Car," Sgcarmart. [Online]. Available: https://www.sgcarmart.com/articles/advice/understanding-omv-parf-and-coe-rebates-for-your-car-30522. [Accessed: Nov. 14, 2024].

VI. Appendix

A. Contribution

- Cheng Yaoyu (e1373768): EDA, Data Cleaning, Feature Engineering, Trained Tree Models, NN models, and the ResNet models, Evaluation, Conducted Experiments, Finetuning, Wrote the report.
- Chen Huaxun (e1351619): Data Cleaning, Feature Engineering, Trained the linear & lasso models, Evaluation, Conducted Experiments, Wrote the report.
- Wang Sen (e1351109): Data Cleaning, Feature Engineering, Wrote the report.
- Hu Senyao (e1351207): Data Cleaning, Feature Engineering, Wrote the report.

B. Additional Missing Values

Figures 8 to 11 show the proportion of missing values for some key features.

C. Price Distribution Fit Pictures

Figures 12 to 20 display the different distribution fits that were attempted to model the target variable.

D. Distribution for different features

Figures 22 to 35 show the distributions for some key features.

E. Additional Count Plots

Figure 36 and Figure 40 shows the occurences for `make` and `model` features.

F. Model Analysis

1) Boxplot of Price by Model: Figure 38 presents a boxplot of car prices grouped by *model* for those with more than 50 occurrences. The variation in median prices across different models within the same make highlights the importance of model-specific characteristics (e.g., engine capacity, features, luxury components) in determining car prices.

2) Violin Plot of Price by Model: The violin plot in Figure 39 visualizes the distribution of prices for each *model*. Models from luxury makes tend to have a broad price range, indicating that features or configurations significantly influence the car's price.

G. Count Plot for Model

Figure 40 shows the frequency of each *model* with more than 50 occurrences in the dataset, providing an overview of the distribution of different car models.

H. Analysis of Model Feature

Figure 41 shows the Bar Plot of mean absolute SHAP values for each feature.

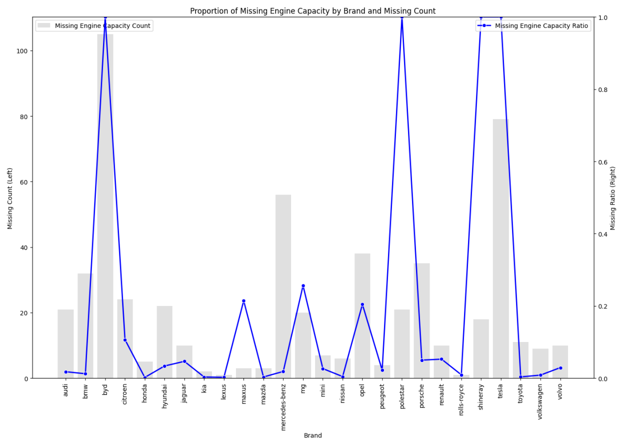


Fig. 8. Proportion of Missing Engine Capacity by Brand

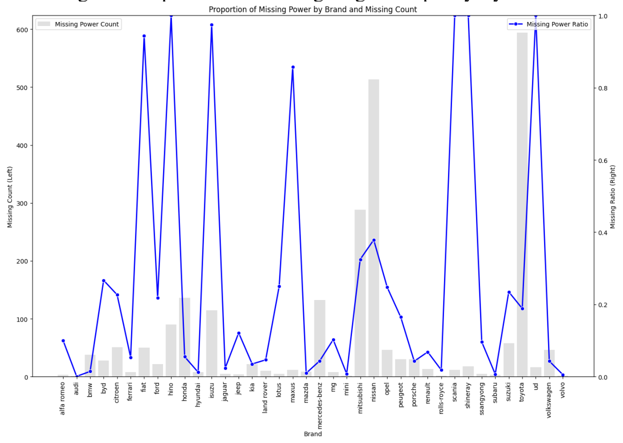


Fig. 9. Proportion of Missing Power by Brand

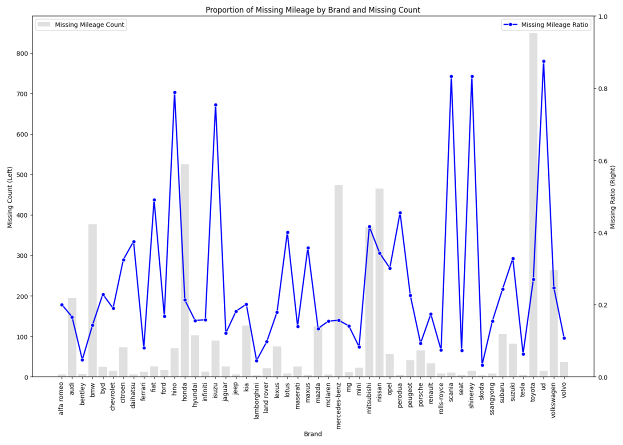
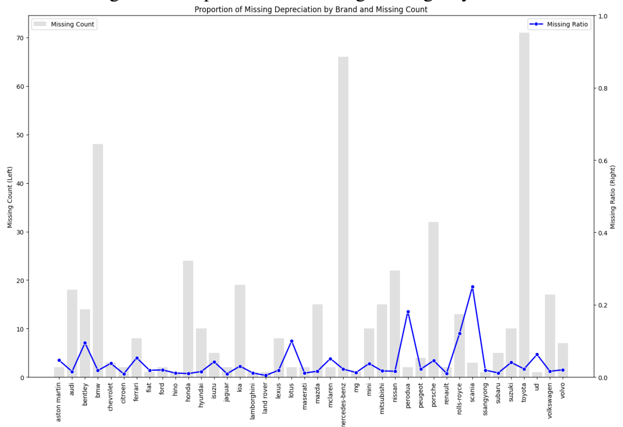


Fig. 10. Proportion of Missing Mileage by Brand



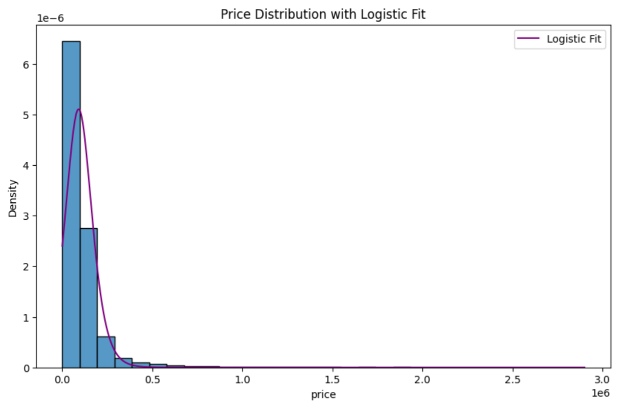


Fig. 12. Price Distribution with Logistic Fit

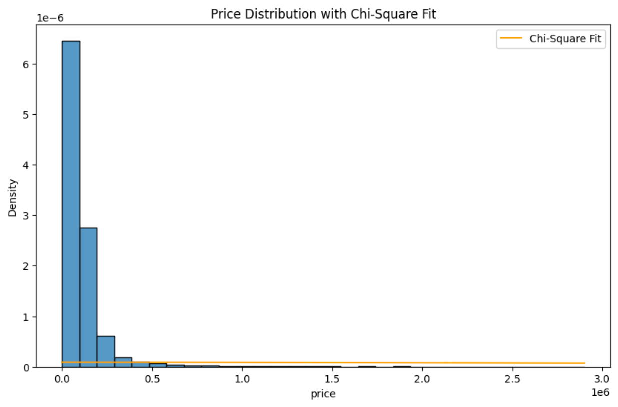


Fig. 13. Price Distribution with Chi-Square Fit

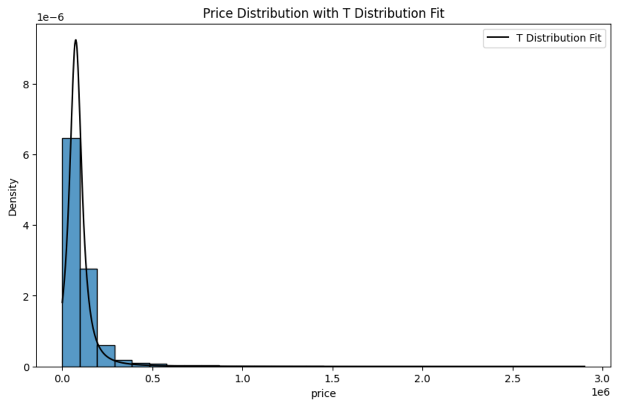


Fig. 14. Price Distribution with T Distribution Fit

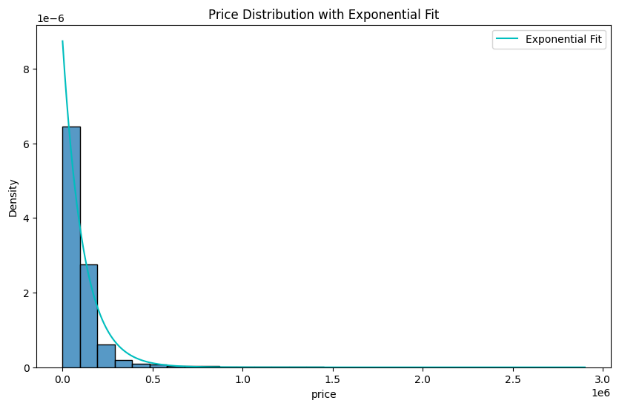


Fig. 15. Price Distribution with Exponential Fit

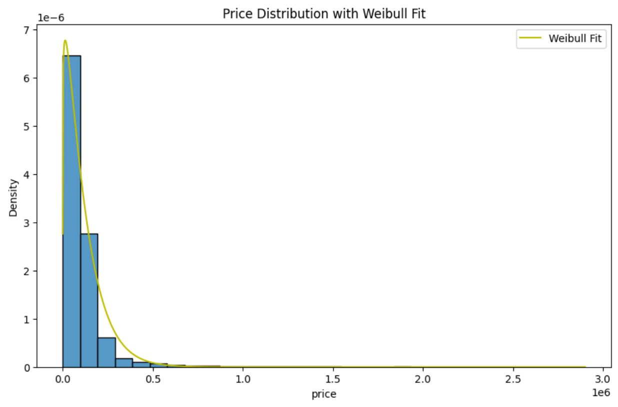


Fig. 16. Price Distribution with Weibull Fit

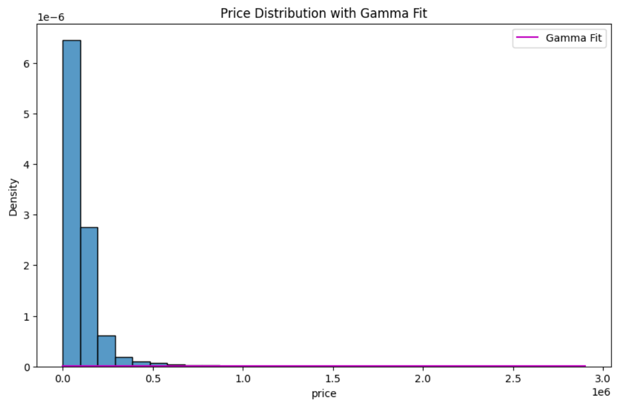


Fig. 17. Price Distribution with Gamma Fit

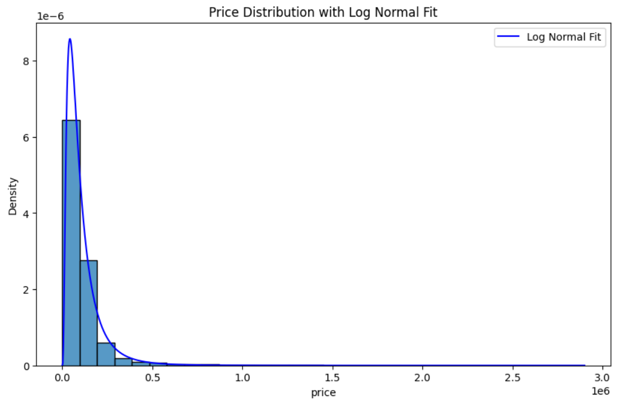


Fig. 18. Price Distribution with Log Normal Fit

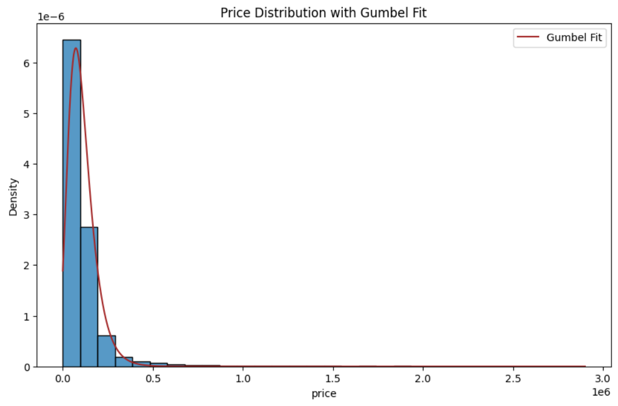


Fig. 21. Price Distribution with Gumbel Fit

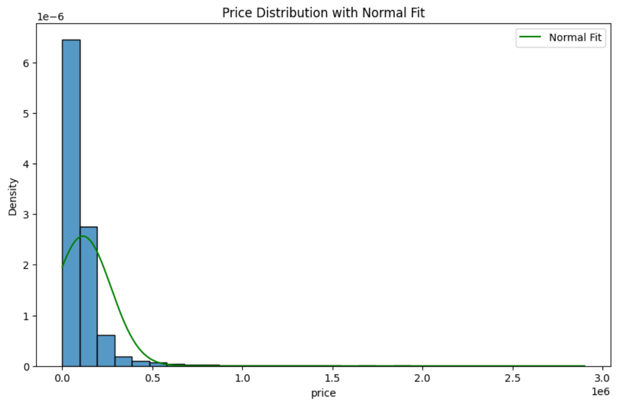


Fig. 19. Price Distribution with Normal Fit

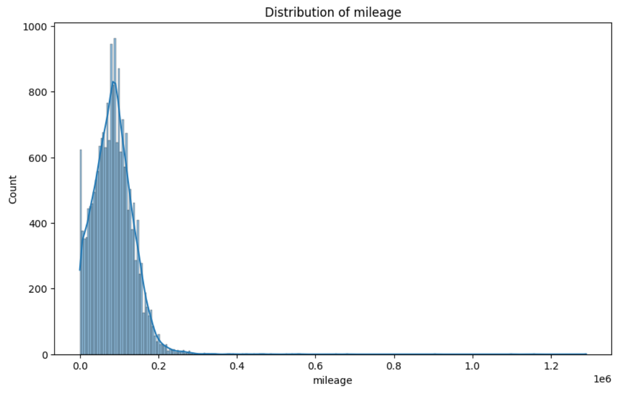


Fig. 22. Distribution of Mileage

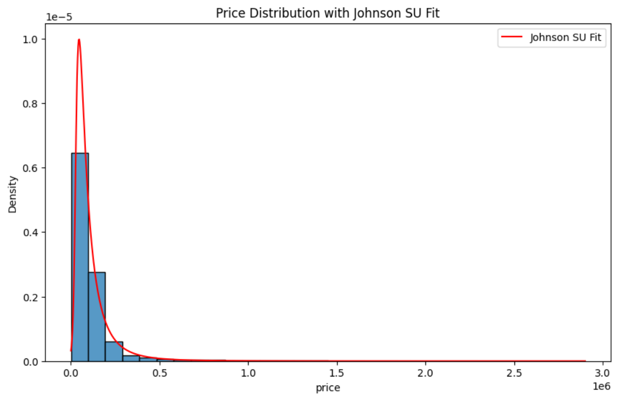


Fig. 20. Price Distribution with Johnson SU Fit

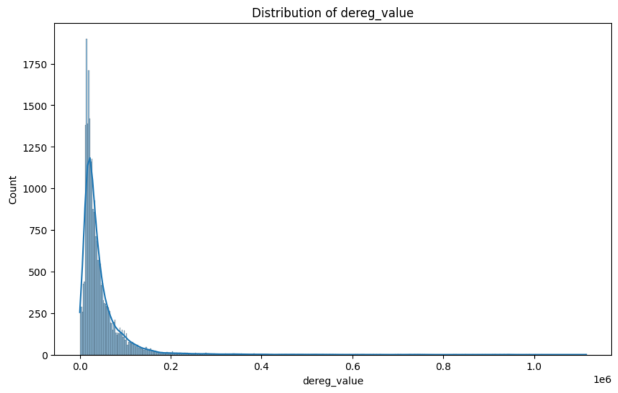


Fig. 23. Distribution of Deregistration Value

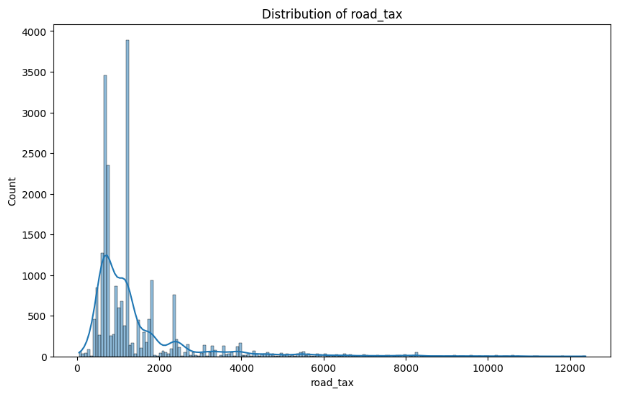


Fig. 24. Distribution of Road Tax

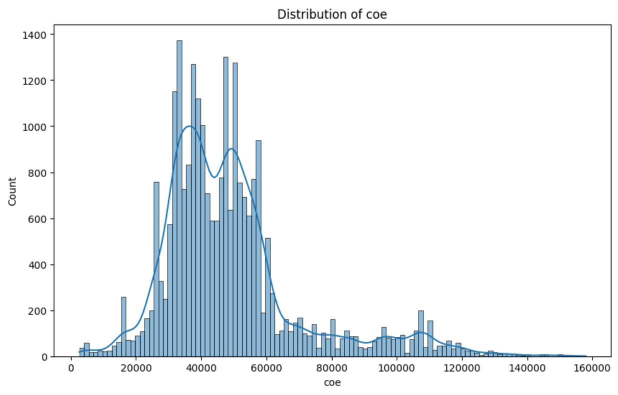


Fig. 25. Distribution of COE

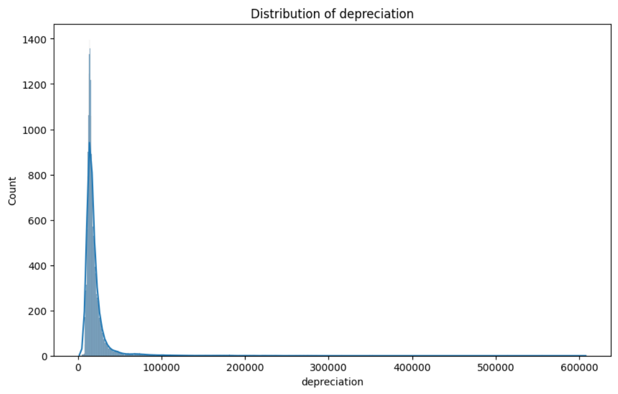


Fig. 26. Distribution of Depreciation

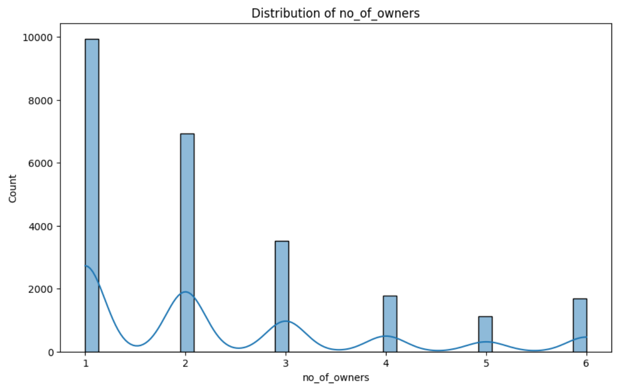


Fig. 27. Distribution of Number of Owners

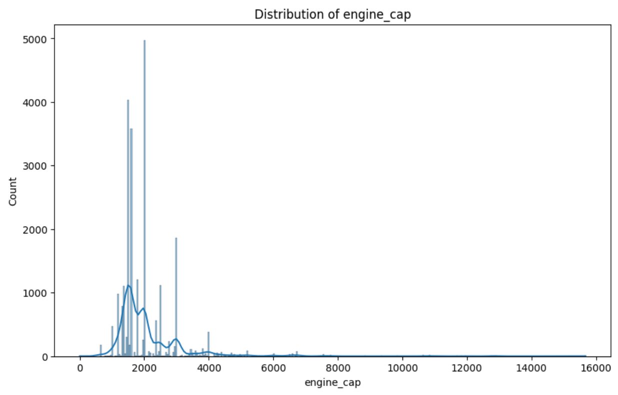


Fig. 28. Distribution of Engine Capacity

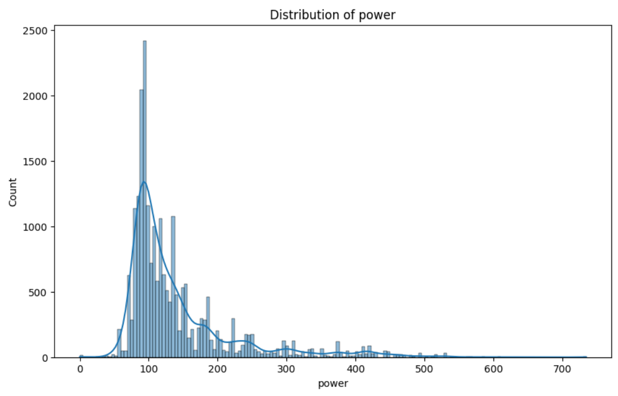


Fig. 29. Distribution of Power

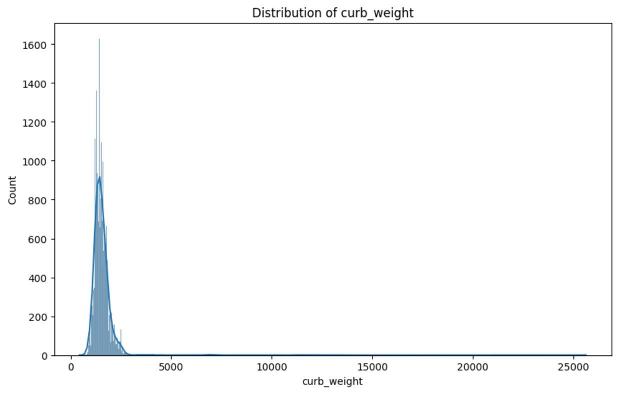


Fig. 30. Distribution of Curb Weight

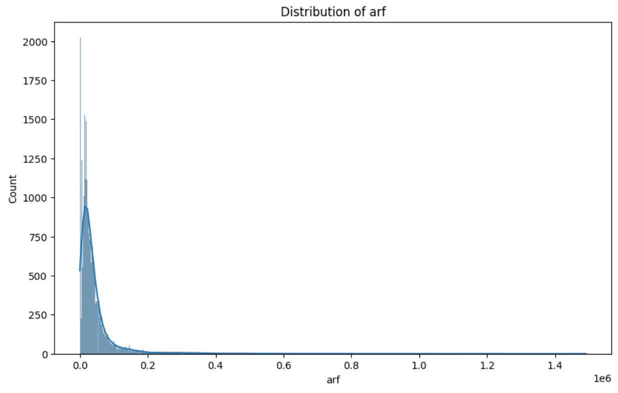


Fig. 33. Distribution of ARF

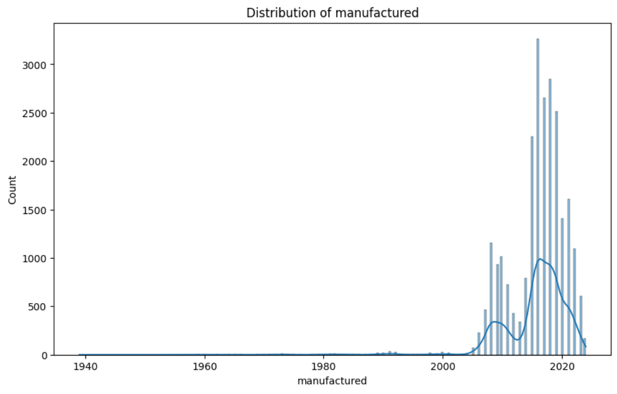


Fig. 31. Distribution of Manufactured Year

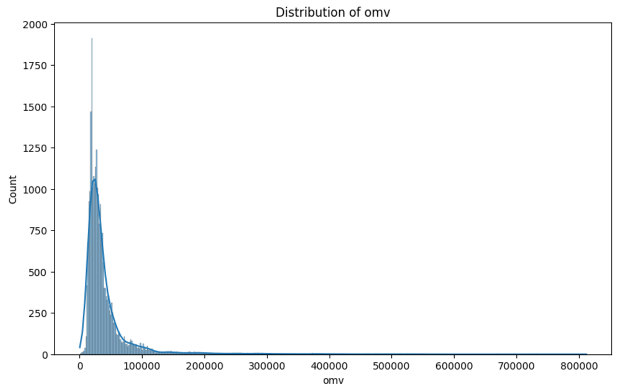


Fig. 34. Distribution of OMV

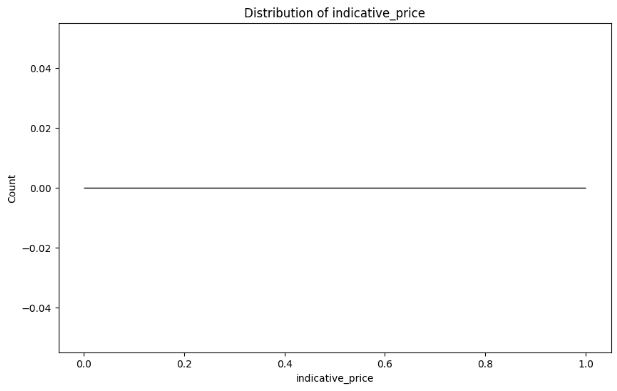


Fig. 32. Distribution of Indicative Price

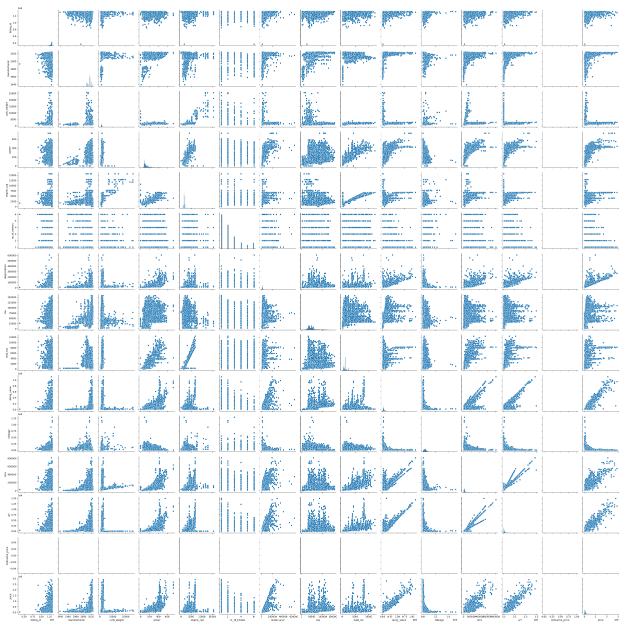


Fig. 35. Pairplot of All Numerical Features

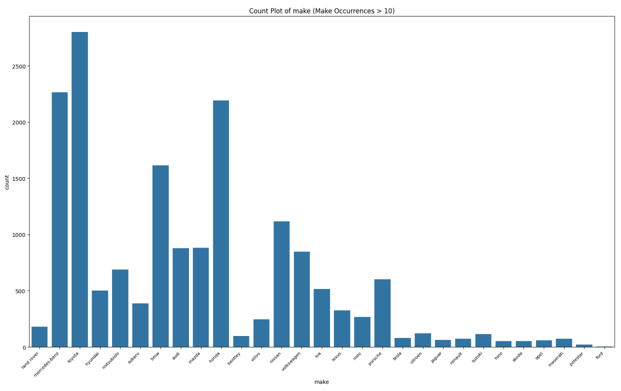


Fig. 36. Count Plot of Make (Make Occurrences >10)

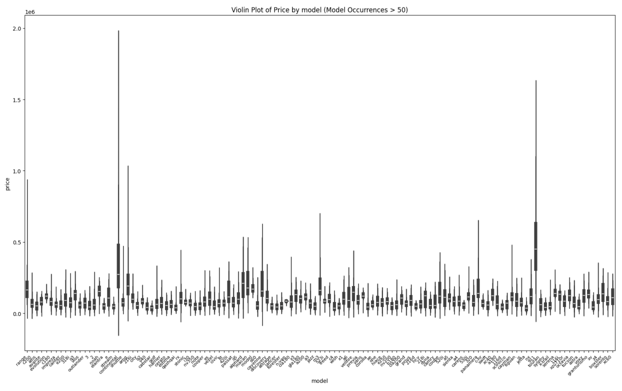


Fig. 39. Violin Plot of Price by Model (Model Occurrences >50)

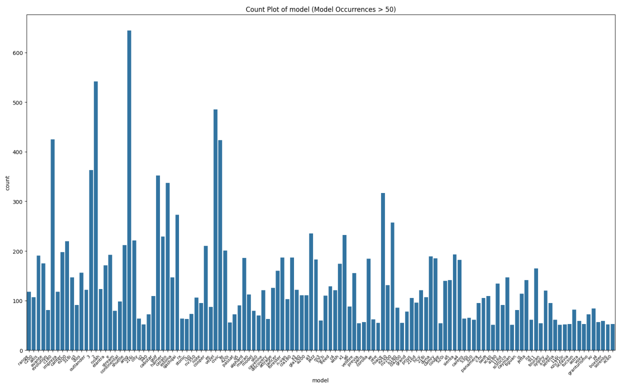


Fig. 40. Count Plot of Model (Model Occurrences >50)

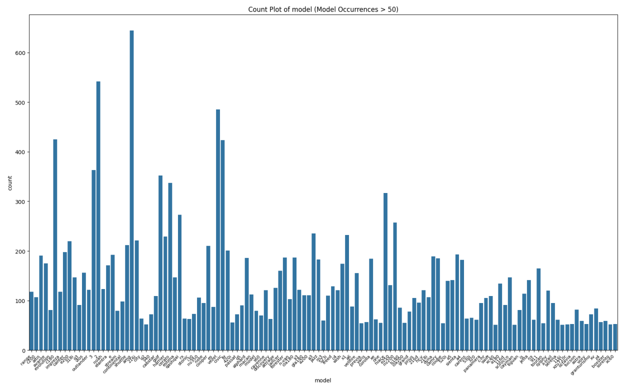


Fig. 37. Count Plot of Model (Model Occurrences >50)

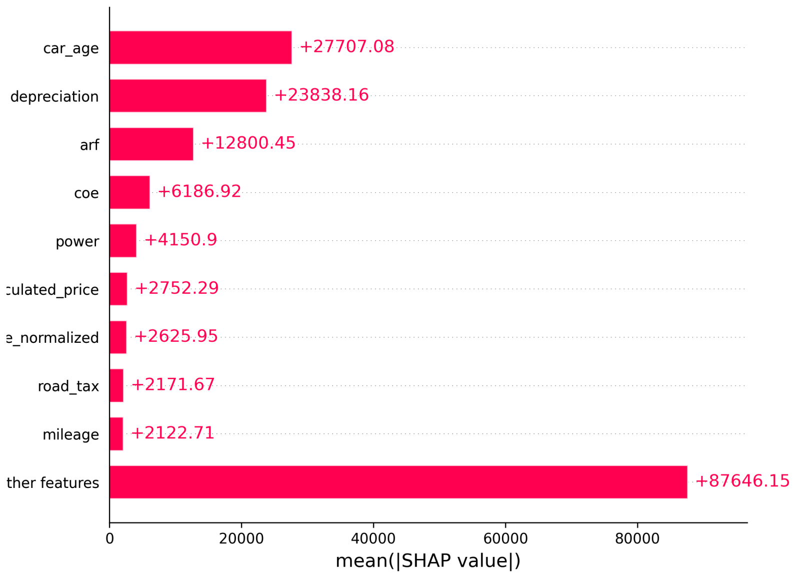


Fig. 41. Bar Plot of Mean Absolute SHAP Values for Each Feature

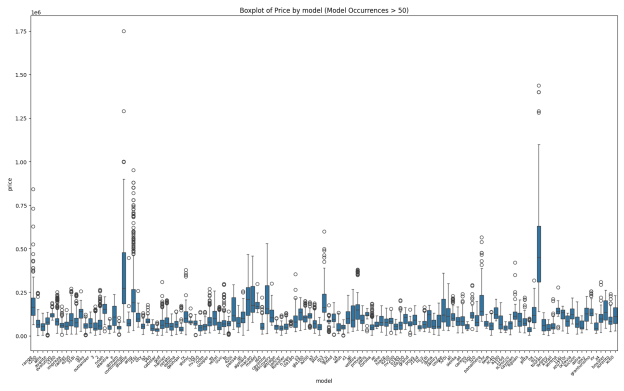


Fig. 38. Boxplot of Price by Model (Model Occurrences >50)