From Data to Deals: A Machine Learning Approach to Used Car Price Prediction

Shiyu Liu

Brown University Data Science Institute

Github repository: https://github.com/lshiyu4210/used_car_price_prediction

1. Introduction

This semester, I decided to buy a car. Given the popularity of purchasing used cars in the United States and my desire to find one I like within a limited budget, I opted to buy a used car. While browsing cars online, I noticed that each car was tagged as an "excellent value," "good value," or "fair value." I believe these auto-trading platforms generate price estimates for each car based on its condition and features, using those estimates to assign the appropriate tags. To make a reasonable estimate myself, I decided to build a machine learning pipeline using techniques learned this semester. I sourced a used car price prediction dataset from Kaggle (Najiib, 2023), which contains 4009 rows and 10 columns, including numerical features like model year, mileage, and price, and categorical features like brand, model, fuel type, and accident history. The target variable is price, with the rest as predictors. Previous studies used models such as XGBoost and Random Forest, with the best results achieving R^2=0.973, RMSE=0.145 (Yılmaz & Selvi, 2023) and MAE = 0.1284 with 87.16% accuracy (Valarmathi et al., 2023). Building on this, I aim to develop a pipeline to predict car prices based on these metrics.

2. Explanatory Data Analysis

At the start of my Exploratory Data Analysis, I examined the distribution of the target variable, "price." Figure 2.1 reveals that the used car sales prices are highly right-skewed (skewness: 19.51), driven by a few luxury cars with significantly higher prices than average. To address this skewness, I applied a log-10 transformation, reducing the skewness to 0.11 and normalizing the distribution, as shown in Figure 2.2. This transformation makes the data more suitable for statistical modeling by mitigating the impact of extreme outliers.

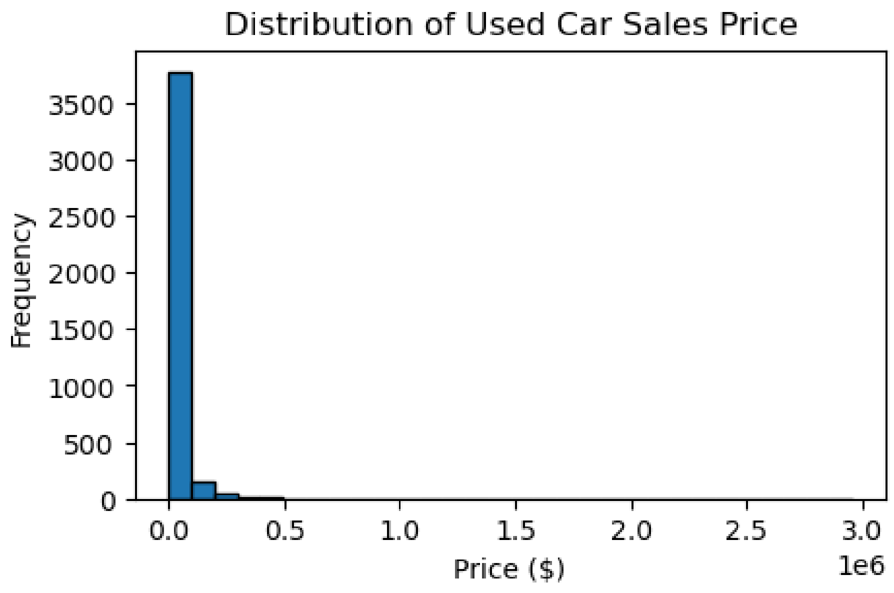


Figure 2.1: Distribution of Used Car Sales Pric

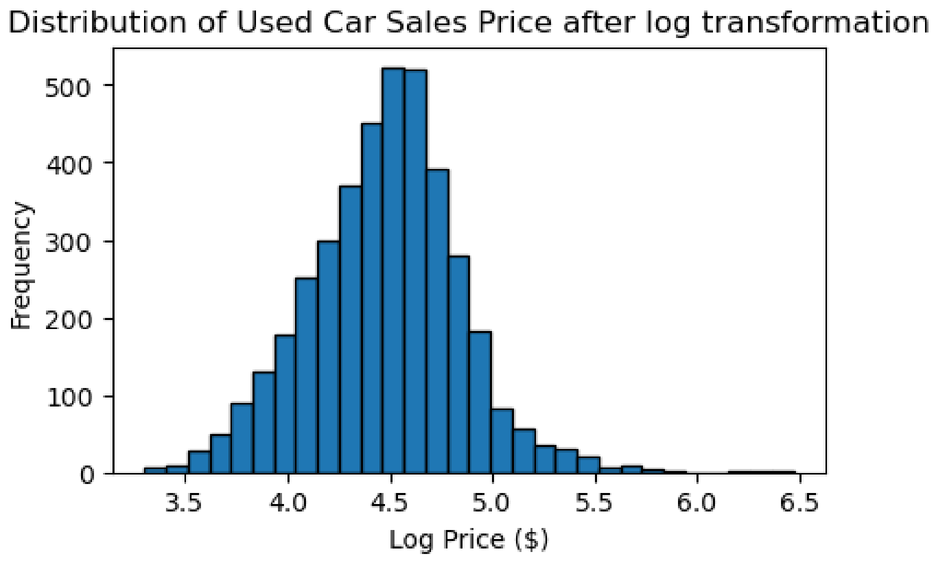
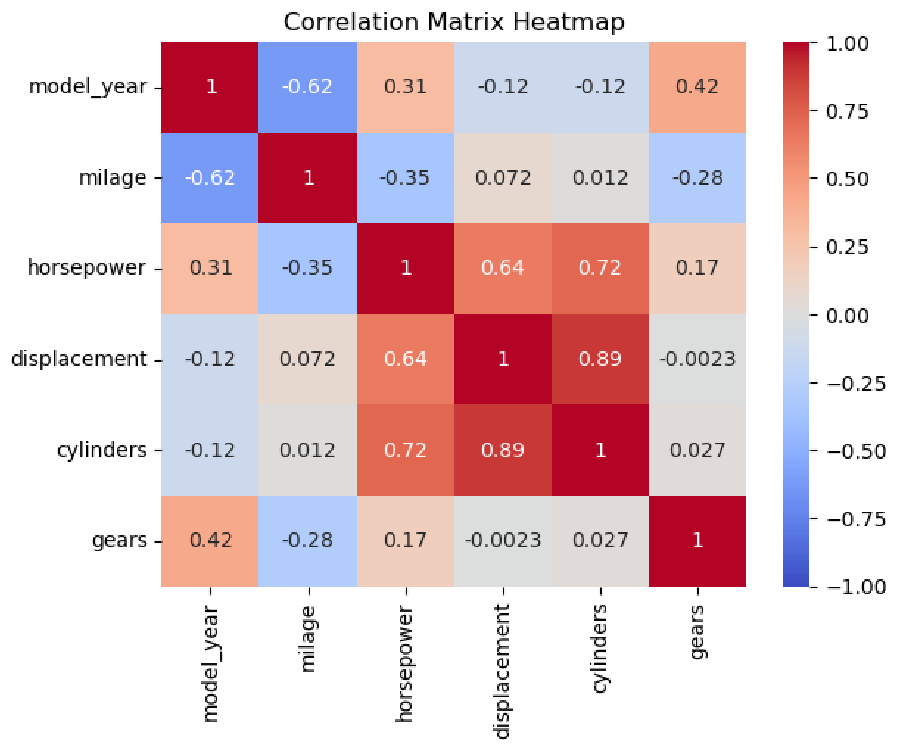


Figure 2.2: Distribution of log-10 Transformed Used Car Sales Prices.

To further investigate, I examined the relationships between numerical variables using a correlation heatmap (Figure 2.3). Features related to engine performance—horsepower, displacement, and cylinders—show strong positive correlations with each other, which aligns with expectations since higher cylinder counts typically result in larger displacement and greater horsepower. Additionally, mileage and model year demonstrate a significant negative correlation, indicating that older cars tend to have higher mileage, which is intuitive.



	model_year	milage	horsepower	displacement	cylinders	gears
model_year	1	-0.62	0.31	-0.12	-0.12	0.42
milage	-0.62	1	-0.35	0.072	0.012	-0.28
horsepower	0.31	-0.35	1	0.64	0.72	0.17
displacement	-0.12	0.072	0.64	1	0.89	-0.0023
cylinders	-0.12	0.012	0.72	0.89	1	0.027
gears	0.42	-0.28	0.17	-0.0023	0.027	1

Figure 2.3: correlation matrix heatmap for continuous features

Finally, I analyzed the impact of the categorical variable “fuel type” on prices using boxplots of log-transformed sales prices (Figure 2.4). The median prices for diesel, hybrid, and plug-in hybrid cars are higher than those for gasoline and E85 flex fuel cars. Gasoline cars, however, exhibit the most variability, with numerous high and low outliers, reflecting the broad range of gasoline-powered car models. Hybrid cars also show a few low outliers. These findings suggest that fuel type influences used car prices, with cleaner energy options generally commanding higher prices.

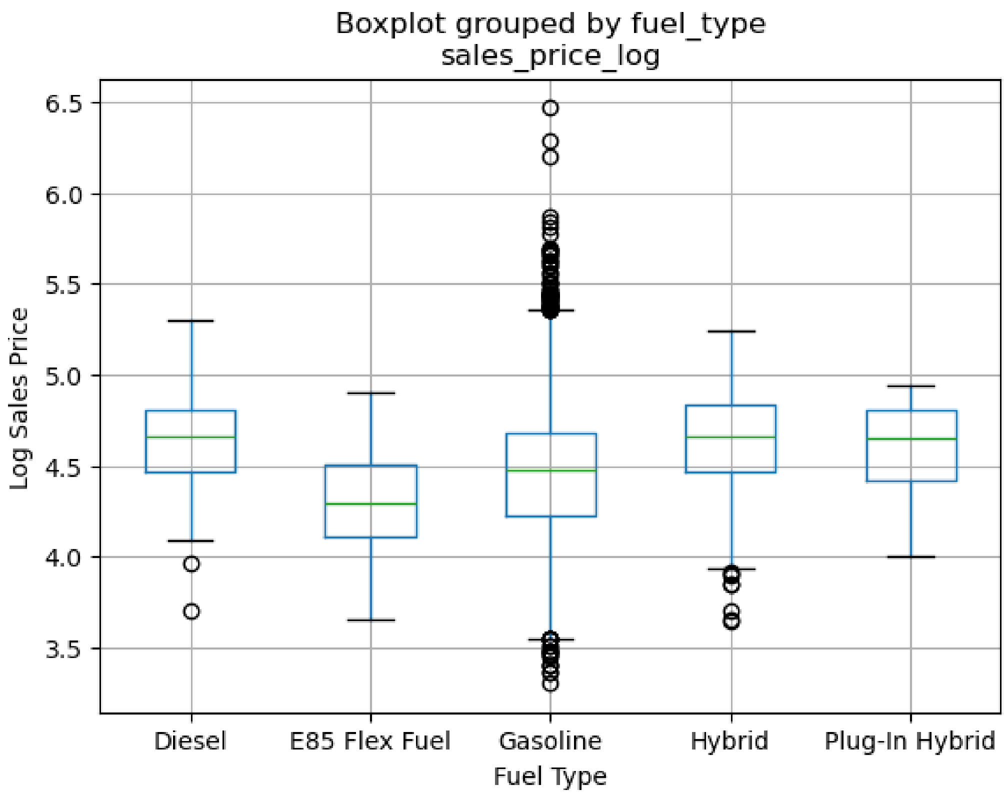


Figure 2.4: Side-by-side Boxplot of Fuel Type vs Log Sales Price

3. Methods

a. Metrics Selection

The reason for using MAE, RMSE, and R^2 in this task is that these metrics complement each other in evaluating model performance. MAE provides the average magnitude of errors in the same unit as the target variable, giving a straightforward interpretation of prediction accuracy. RMSE penalizes large errors, which is crucial for pricing tasks where extreme under- or over-predictions can be costly. R^2 measures the proportion of variance explained by the model, indicating how well the model captures the overall patterns in the data.

b. Feature Engineering

The first step in working with the dataset was to apply feature engineering to the "engine" and "transmission" columns. These categorical variables contained too many distinct categories to be predictive in their current form. From the "engine" column, I derived attributes such as horsepower, engine displacement, the number of cylinders, and whether the engine is turbocharged. Similarly, from the "transmission" column, I extracted whether the transmission is automatic or manual and the number of gears.

c. Pipeline

The data was then processed through a machine learning pipeline. To ensure robustness, I looped through 5 random states: 42, 84, 126, 168 and 210. In each loop, the data was split using train_test_split with a 20% test size, as all data points were i.i.d. Within the training data, I applied 4-fold cross-validation to evaluate model performance and hyperparameter tuning.

A preprocessor was constructed to handle categorical and numerical features. Categorical variables were transformed using OneHotEncoder, while missing numerical values were imputed using a customized XGBImputer, which incorporates multivariate imputation with XGBRegressor (n_estimators=500, learning_rate=0.1, max_depth=5). Since the data is Missing Completely at Random (MCAR), this approach ensures robust handling of missing values. Finally, StandardScaler was applied to normalize the numerical features. This preprocessing was integrated into the pipeline.

Then, I constructed a general parameter grid (Table 3.1) covering models such as Lasso, Ridge, ElasticNet, RandomForestRegressor, KNeighborsRegressor, and XGBRegressor. The preprocessor, 4-fold cross-validation, and parameter grid were passed through GridSearchCV to tune hyperparameters for each random state. I recorded the mean and standard deviation of MAE, RMSE, and R^2 across the random states.

The results showed that XGBRegressor outperformed all other models across the three metrics. To further improve its performance, I expanded the parameter grid (Table 3.2 and constructed a new pipeline, removing the imputation step since XGBRegressor handles missing values natively. I then retrained the model with the updated configuration to optimize its performance. To ensure reproducibility, I fixed random states in all steps where randomness was involved, including splitting, imputation, and model training.

Model Name	**Parameters**
Lasso	alpha: [0.01, 0.1, 1, 10, 100]
Ridge	alpha: [0.01, 0.1, 1, 10, 100]
ElasticNet	alpha: [0.01, 0.1, 1, 10, 100]
	l1_ratio: [0.2, 0.4, 0.6, 0.8]
RandomForestRegressor	max_depth: [1, 3, 10, 30, 100]
	max_features: [0.25, 0.5, 0.75, 1.0]
KNeighborsRegressor	n_neighbors: [3, 5, 10, 20]
	weights: ['uniform', 'distance']
XGBRegressor	n_estimators: [100, 200, 500, 1000]
	learning_rate: [0.1, 0.2, 0.5, 1.0]
	max_depth: [3, 5, 10]

Table 3.1: General Parameter Grid for Model Training

Model Name	**Parameters**
XGBRegressor	n_estimators: [100, 300, 500, 700, 1000]
	learning_rate: [0.01, 0.1, 0.2, 0.5]
	max_depth: [3, 5, 7, 10]

Table 3.2: Expanded Parameter Grid for XGBoost Regressor Fine-tuning

4. Results

a. Machine Learning Model Performance Comparison

My baseline prediction is calculated by predicting every entry as the mean of **y_test**. This results in baseline performance metrics of MAE: 0.284, RMSE: 0.358, and R^2: 0. The mean, standard deviation, and the number of standard deviations above the baseline for MAE, RMSE, and R^2 are presented in Figures 4.1, 4.2, 4.3 and Table 4.1, respectively.

Based on the performance metrics across three figures, XGBoost Regressor consistently outperforms the other models. In terms of Mean Absolute Error (MAE), XGBoost achieves the lowest value of 0.0875, accompanied by the smallest standard deviation (0.0022) and the highest z-score of 87.835, indicating its robustness and accuracy. Similarly, for the Root Mean Square

Error (RMSE), XGBoost also delivers the best performance with a mean RMSE of 0.1343 and a z-score of 15.923, though its standard deviation is relatively higher (0.0140). When evaluating the models using the coefficient of determination (R^2), XGBoost again stands out with the highest mean R^2 of 0.8721, signifying its outstanding ability to explain the variance in the data. Random Forest Regressor and KNeighbors Regressor follow closely, performing relatively well across all metrics. However, linear models such as Lasso, Ridge, and ElasticNet generally lag behind, with higher error values and lower R^2 scores. Ridge regression performs slightly better among the linear models, achieving a moderate balance between RMSE and R^2 metrics. Overall, XGBoost demonstrates the most reliable and accurate performance across MAE, RMSE, and R^2, making it the top-performing model.

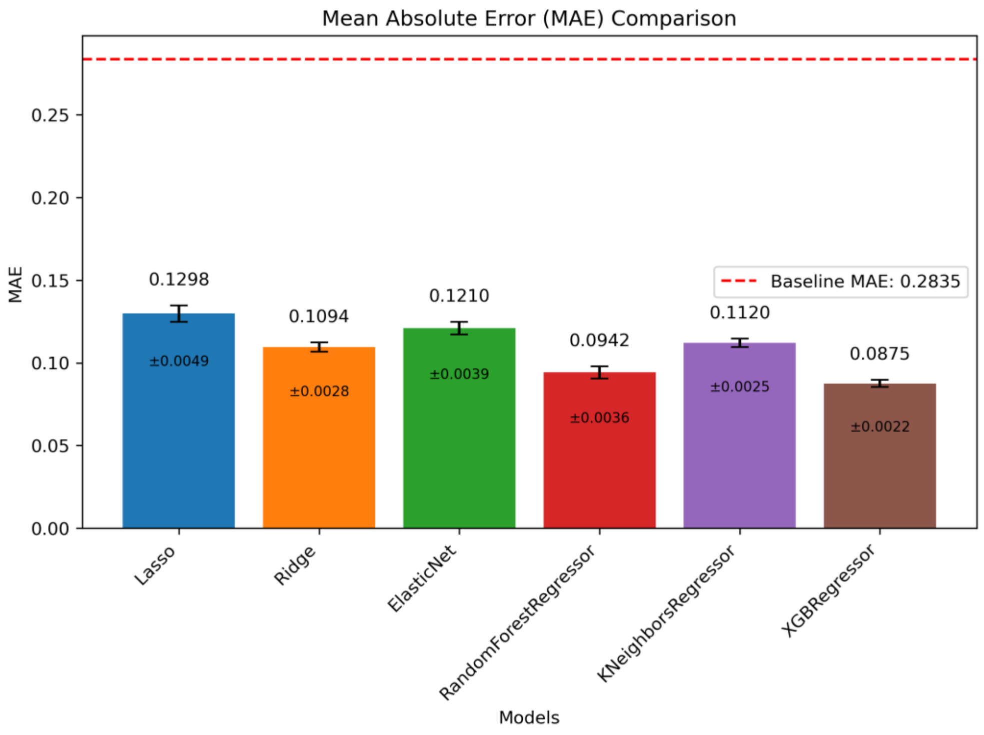


Figure 4.1: Model Performance Summary for MAE

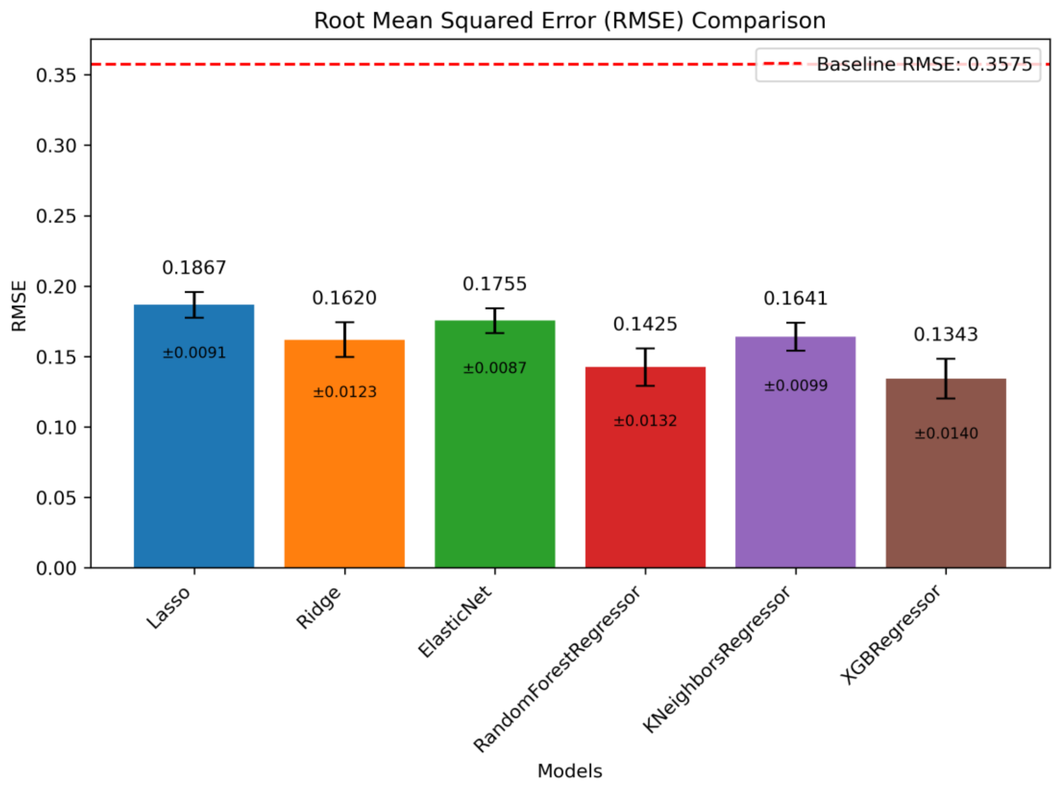


Figure 4.2: Model Performance Summary for RMSE

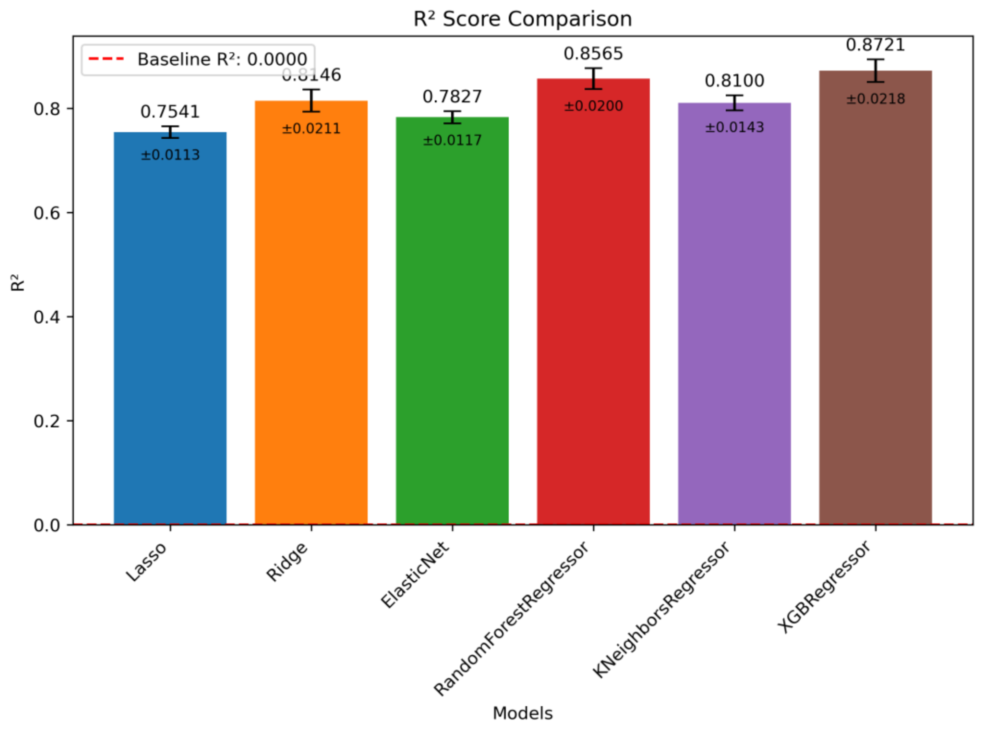


Figure 4.3: Model Performance Summary for R^2

Model Name	**MAE z score**	**RMSE z score**	**R^2 z score**
Lasso	31.220	18.799	66.616
Ridge	62.239	15.926	38.542
ElasticNet	41.511	20.838	66.974
RandomForestRegressor	52.295	16.243	42.823
KNeighborsRegressor	68.785	19.561	56.550
XGBRegressor	87.835	15.923	40.075

Table 4.1: Z Scores for Three Metrics

b. Fine-tuned XGBoost Regressor Performance

The fine-tuned XGBoost Regressors demonstrate exceptional performance across metrics (Table 4.2). The model optimized for MAE achieves a best score of 0.0854 with a smaller max_depth of 3, indicating that a simpler model is sufficient for minimizing absolute errors. In contrast, the models optimized for RMSE (0.1107) and R^2 (0.9041) use a deeper tree (max_depth = 5), suggesting that greater model complexity helps capture variance and improve overall predictive performance for these metrics.

Metric	**Best Score**	**n_estimators**	**learning_rate**	**max_depth**
MAE	0.0854	1000	0.1	3
RMSE	0.1107	1000	0.1	5
R^2	0.9041	1000	0.1	5

Table 4.2: Best XGBRegressor for Each Metric

c. Global Importance

To better understand which features most influence the model's performance, I analyzed the global importance of my best-performing XGBRegressor (in terms of RMSE and R^2) using permutation importance, as well as the gain and weight metrics.

The permutation importance plot shows that mileage, model_year, and horsepower are the most influential features, significantly impacting the model's performance. In contrast, variables like fuel_type and transmission_type have minimal influence. Car buyers and sellers should prioritize mileage, car age, and horsepower, as these factors heavily influence market value.

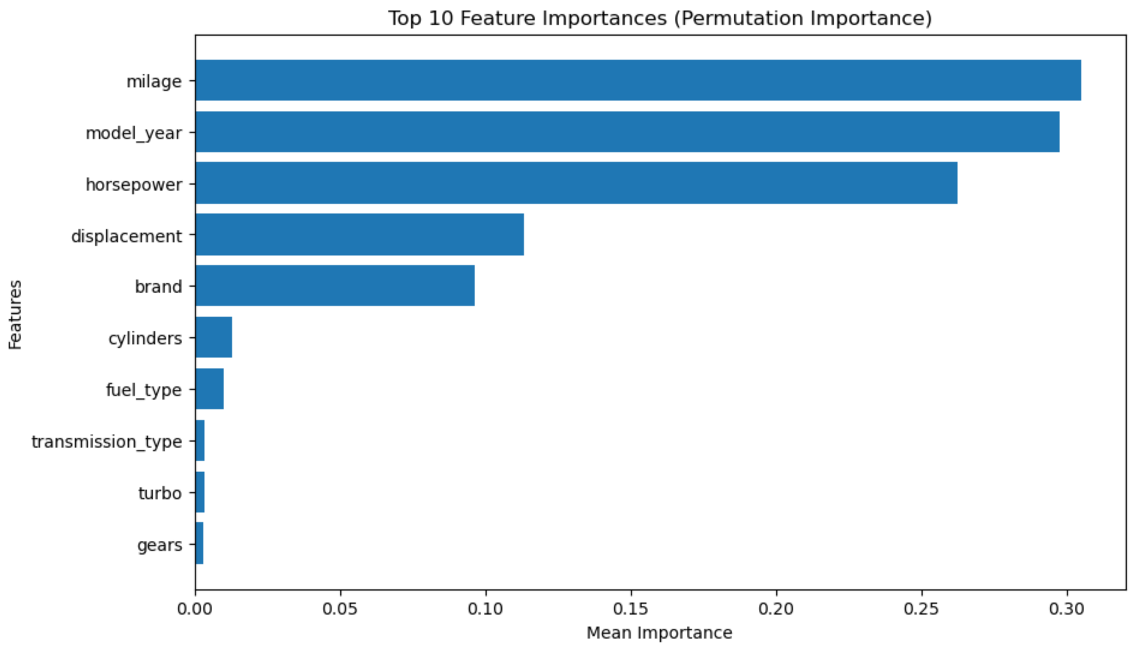


Figure 4.4: Permutation importance of the top 10 features for the XGBRegressor.

In terms of gain (Figure 4.5), mileage provides the largest improvement to the model's accuracy, followed by luxury brands like Lamborghini and Porsche, highlighting their strong contribution to reducing prediction loss. Key numerical features such as horsepower and displacement also play a significant role, emphasizing the importance of these factors in general car valuation.

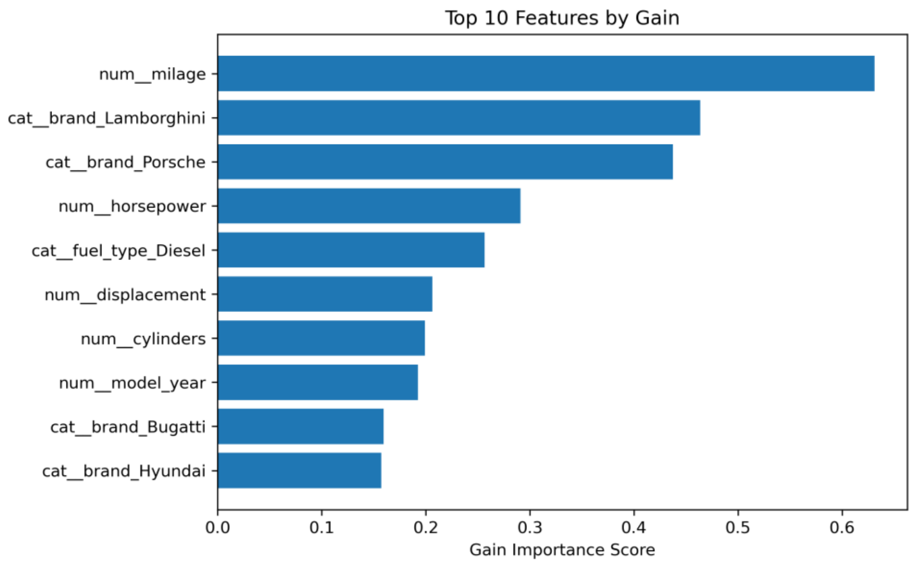


Figure 4.5: Top 10 features by Gain.

The weight plot indicates that model_year and mileage are the most frequently used features for decision splits, followed by horsepower and displacement. Categorical features like interior color and accident history are less frequently used but still contribute to predictions. The frequent use of mileage and model year suggests that consistent maintenance records and car age transparency are crucial for accurately assessing used car prices.

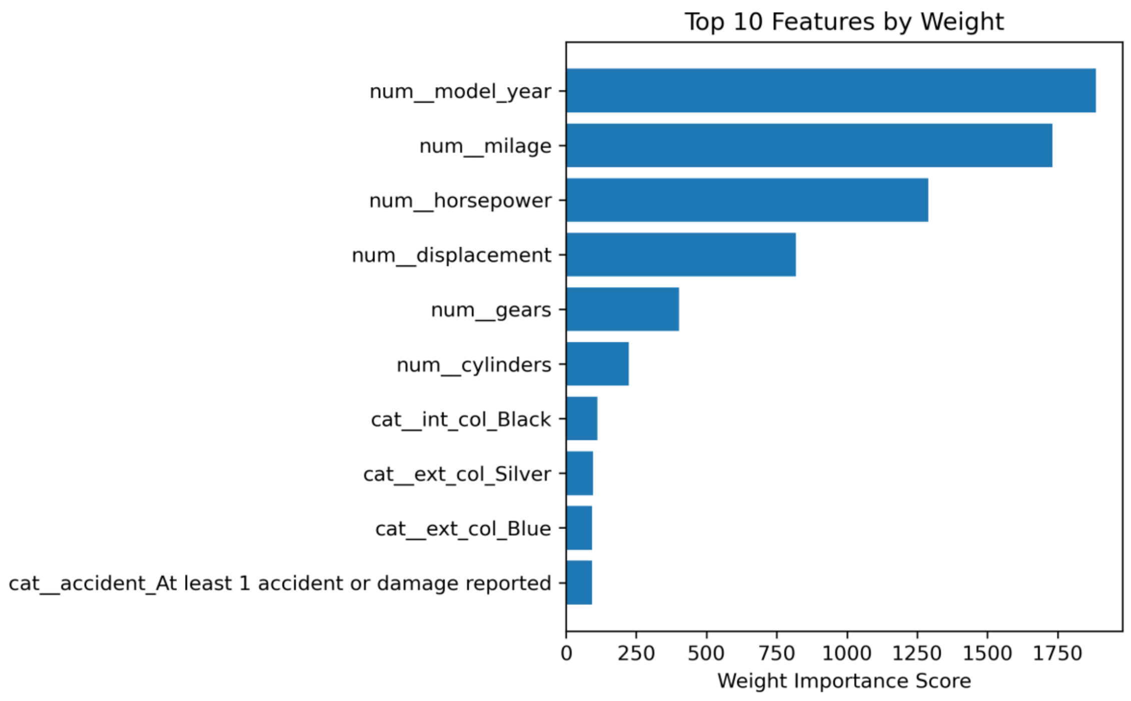


Figure 4.6: Top 10 features by Weight.

d. Local Importance

Two points with the highest and lowest log sales price are selected for local interpretation. Figure 4.7 shows that features like model year (0.57), displacement (1.99), and luxury brand Lamborghini (1.0) strongly push the prediction upward, leading to the highest log sales price of 5.73. In contrast, mileage (-1.1) slightly reduces the prediction, reflecting the expected depreciation effect of higher mileage.

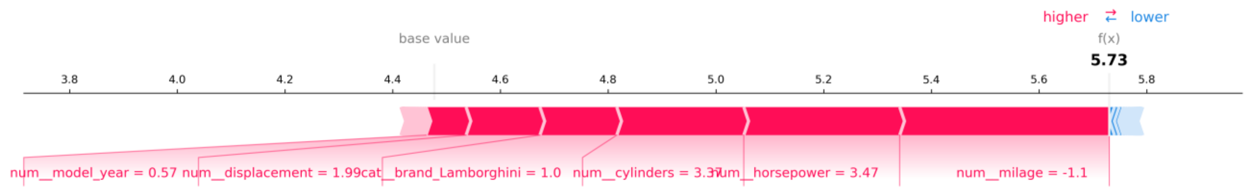


Figure 4.7: SHAP Force Plot for Highest Log Price

For the lowest log sales price of 3.78 (Figure 4.8), features like horsepower (-1.48), mileage (1.5), and model year (-1.08) pull the prediction downward, indicating a weaker vehicle condition or older age. Additionally, the brand Mitsubishi and automatic transmission contribute slightly to increasing the price but are insufficient to offset the downward forces.

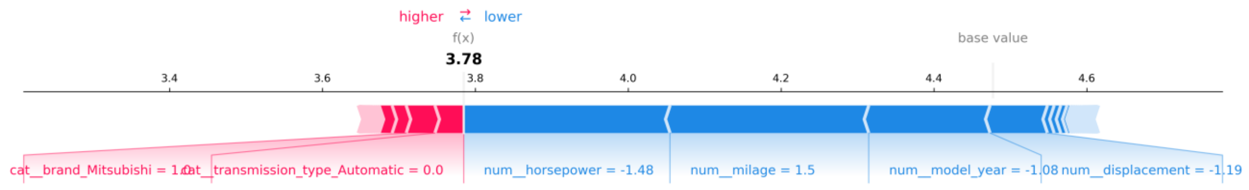


Figure 4.8: SHAP Force Plot for Lowest Log Price

5. Outlook

This project has several areas for future improvement. To enhance predictive performance, a reduced feature method could replace the imputation process, as it would mitigate the uncertainty introduced by imputing missing values, particularly when feature correlations are weak. Additionally, more advanced models like CatBoost (Valarmathi et al., 2023) and LightGBM can be explored for better accuracy and efficiency. Incorporating Lasso regression or PCA for feature selection (Yılmaz & Selvi, 2023) could further reduce model runtime and improve overall performance.

For interpretability, SHAP summary plots can help explain global feature importance and their impact on predictions, while LIME can provide localized explanations by analyzing individual predictions.

A key limitation of this project is the high cardinality of features such as brand, model, and exterior/interior color, which generates numerous columns, slowing down processes like multivariate imputation. A possible solution is to group low-frequency categories into an "other" category or use target encoding for high-cardinality features.

Finally, collecting a larger dataset with more data points and including additional features like fuel consumption could improve model robustness and accuracy.

6. References

Najiib, T. (2023). *Used Car Price Prediction Dataset* [Data set]. Kaggle. https://www.kaggle.com/datasets/taeefnajib/used-car-price-prediction-dataset

Valarmathi, B., Srinivasa Gupta, N., Santhi, K., Chellatamilan, T., Kavitha, A., Raahil, A., & Padmavathy, N. (2023). *Price estimation of used cars using machine learning algorithms*. Vellore Institute of Technology, Tamil Nadu, India.

YILMAZ, Seda, & SELVİ, İhsan Hakan. (2023). Price Prediction Using Web Scraping and Machine Learning Algorithms in the Used Car Market. *Sakarya University Journal of Computer and Information Sciences*, *6*(2), 140–148. https://doi.org/10.35377/saucis...1309103