Final Project Report

Project Title: *AutoHaggle*

Maalolan Bharaniraj, Jack Carpini, Xiaoyang Fei
Supervisor: Dr. Fatema Nafa
Northeastern University / DS 5500

April 15, 2025

Contents

Abstract

AutoHaggle addresses the challenge of determining fair vehicle pricing in the dynamic automotive market. We developed a machine learning solution that provides accurate price predictions for used vehicles by creating a clustering-based model pipeline. Using data from over 680,000 car transactions, we preprocessed features, addressed inflation effects, and trained multiple regression models on distinct vehicle clusters. The best-performing approach used column-wise inflation adjustment with LightGBM models, achieving R^2 values up to 0.97 and deployable RMSE metrics. Our web application provides users with fair market price estimates, confidence ranges, and visual comparisons, enabling better purchasing decisions for consumers and optimized pricing strategies for businesses.

1 Introduction

In the automotive industry, determining whether a vehicle price quote is fair is a challenging task due to the dynamic nature of car pricing. Variations in location, timeframe, vehicle features, and market conditions contribute to inconsistencies in pricing. These disparities make it difficult for buyers to assess whether they are getting a good deal and for businesses to optimize pricing strategies effectively. Our team aims to minimize these inconsistencies by developing a fair, accurate, and dynamic vehicle pricing tool to help consumers and businesses more effectively price used vehicles. Our objectives for this project were to initially find enough high quality data to use for modeling, clean and prepare the data and features for modeling, cluster the data into similar transactions, train individual models on each cluster, and finally deploy our pipeline to the web as an easy to use tool.

To make our solution accessible and user-friendly, we built a responsive front-end interface that allows users to input relevant vehicle details—such as make, model, year, mileage, location, and features—through a simple web form. Upon submission, these inputs are sent to our FastAPI-powered backend, which hosts the trained machine learning models and processes predictions in real time.

The API is designed to be lightweight and scalable, handling preprocessing, clustering the input into its appropriate segment, and serving the price prediction from the corresponding model. The front-end then displays the predicted fair market price along with a confidence range and visual comparisons to similar listings in the market.

By integrating the front-end with our API, we ensure a seamless end-to-end user experience, turning a complex pricing engine into a clean, intuitive tool for both consumers and industry stakeholders.

2 Related Work

Previous approaches to vehicle price prediction typically rely on single regression models trained on entire datasets without segmentation. Studies have explored linear models, tree-based approaches, and neural networks with varying success. Commercial solutions from companies like Kelley Blue Book and Edmunds provide pricing estimates but lack transparency in their methodologies. Our work builds upon these approaches but introduces clustering-based segmentation and inflation adjustment techniques to improve accuracy across diverse vehicle categories.

3 Methodology

3.1 Data Collection

Both of our datasets come from Kaggle:

- Dataset 1 (558,837 rows, reduced to 12 features).
- Dataset 2 (122,144 rows, reduced to 8 features).

3.2 Data Preprocessing & Feature Engineering

These datasets contain records of car sales from 2014-2015 and 2019-2020, respectively, along with features about each vehicle and the transaction. The original datasets both contained features we felt contained no information relevant for modeling (VIN, ID, etc.) so these were dropped. Each feature was cleaned so that each categorical feature was filled with the mode of that feature and numerical features were filled with the median due to a right skew in all numerical columns. We merged both dataset to increase our total number of records and dropped any records where we could not impute values (missing make/model/trim). We also filled in the state sold column by extracting the state from the zip code provided. Our biggest task in this portion was handling the sale price column to properly account for inflation. We created three datasets from our original merged dataset with different inflation adjustments. Our first solution was to just leave the price in the original year uninflated dollars. Our second option was to do a column wise inflation adjustment and convert the whole column to 2024 dollars. Our final option was to do a row wise inflation adjustment and simulate records with the 2024 price. Our final features for each of our three datasets were: year, make, model, trim, state, condition, odometer, mmr, selling_price, zip code, mileage, price_sold.

3.3 Clustering

Each dataset was clustered to segregate transactions based on their numerical features, creating smaller, more homogeneous groups that could improve modeling performance. The first raw dataset was clustered using KMeans clustering into 3 clusters, based solely on numerical features such as mileage, year, and price. These features were normalized using

Scikit-learn's StandardScaler, ensuring that each feature contributed equally to the clustering process. PCA was later used to visualize the cluster separation and interpret feature influence.

The second column-wise inflation-adjusted dataset followed a similar process, using KMeans clustering on normalized numerical features to create 3 clusters. These clusters served as the basis for training Lasso, Ridge, and LightGBM models.

For the row-wise inflation-adjusted dataset, clustering was again performed with KMeans, following normalization of numeric columns. We compared the model performance across clusters and observed that LightGBM trained on this version achieved the best overall results, benefiting from consistent pricing scales and reduced temporal variance.

3.4 Modeling

To test our datasets we picked 3 different models to experiment with. We wanted to test models of varying complexity so we chose 2 simpler models, Lasso and Ridge regression, and a more complex tree based model, LightGBM. We also attempted to implement a deep learning model but this yielded very poor results and was not pursued any further. Each model was run initially with default parameters to get a baseline score on each cluster and parameter tuning was implemented where necessary. K-Folds cross validation was also used on clusters containing fewer data points to try and create a more robust and generalizable model. All models were evaluated primarily using RMSE as it is easy to interpret in the context of our problem statement with R^2 also being calculated as a secondary performance measure.

4 Experimental Setup

4.1 Datasets

We utilized two Kaggle datasets with car sales data from 2014-2015 and 2019-2020, containing a combined total of approximately 680,000 records after preprocessing.

4.2 Tools and Libraries

- Python for data processing and modeling
- Scikit-learn for preprocessing, clustering, and baseline models
- LightGBM and XGBoost for gradient boosting models
- FastAPI for backend development
- React for frontend development

4.3 Hardware and Environment

Development was performed on local workstations with 16GB RAM and cloud-based virtual machines for larger computations.

4.4 Evaluation Metrics

Our primary evaluation metric was Root Mean Squared Error (RMSE), with the coefficient of determination (R^2) as a secondary metric.

5 Results and Analysis

5.1 Raw Data

For the original raw merged dataset, after our initial runs with each model we began adjusting the clustering and modeling parameters to boost performance. At first we did clustering for k = 3 and our results on our models for the 3 clusters were really bad, most of them resulting in NaN. After playing around with reasonable k's from our elbow method results, k = 5 provided the best results with some of the clusters giving us 90% accuracy, while 2 of them still staying in NaN, and others less than 30% accuracy. We ended up dropping the 2 irrelevant clusters since they carried little to no information and their size was extremely small. We moved forward with 3 clusters. Adjusting lasso regression made no difference so we kept it as is. But in conclusion, the raw dataset contained cars sold in different years with their original sale prices, which made direct comparisons between old and new transactions unreliable. We wanted to find a viable approach to modify the dataset, so not only the modeling results stay good or get better but also the data that's contained is useful and insightful for car quote prediction.

Test	RMSE	R^2
Cluster: 0 Lasso Regression	7412.82	.29
Cluster: 0 Ridge Regression	7412.82	.29
Cluster: 1 Lasso Regression	1737.58	.97
Cluster: 1 Ridge Regression	1737.58	.97
Cluster: 4 Lasso Regression	9866.98	-0.37
Cluster: 4 Ridge Regression	9764.99	-0.34
Cluster: 0 LightGBM	5133.30	.66
Cluster: 1 LightGBM	1733.18	.97
Cluster: 4 LightGBM	8388.16	.01

Table 1: Model Performance on Raw Dataset

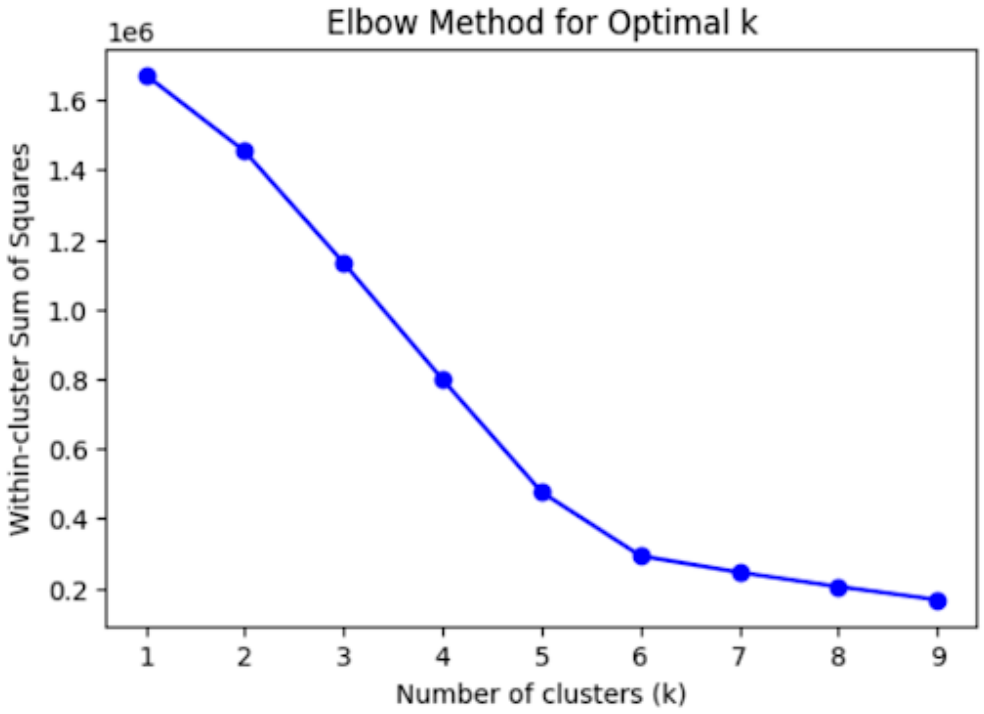

Figure 1: Elbow Method for Optimal k

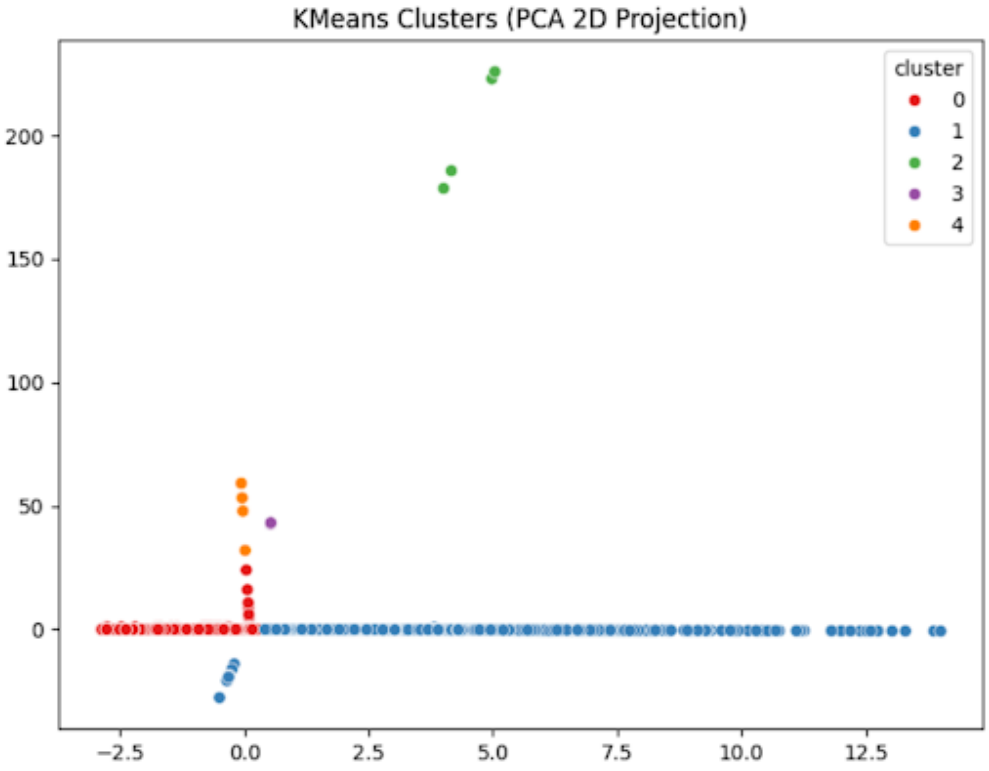

Figure 2: LGBM Cluster 0 Feature Importance

5.2 Column Wise Inflation Adjustment

For the column-wise inflation adjusted dataset, simple KMeans clustering was used to cluster the data into 3 clusters of sizes 136622, 461080, 71190, respectively. Using Lasso and Ridge Regression with the base parameters we were able to achieve the following results on each cluster.

Cluster/Model	RMSE	R^2
0 / Lasso	2737.073	0.954
0 / Ridge	2737.080	0.954
1 / Lasso	8894.405	0.250
1 / Ridge	8894.405	0.250
2 / Lasso	3044.192	0.942
2 / Ridge	3044.189	0.942

Table 2: Linear Model Performance on Column-wise Inflation Adjusted Dataset

Training a LightGBM model with base parameters on each cluster yielded the following results.

Cluster	RMSE	R^2
0	2774.97	0.952
1	5630.748	0.699
2	2229.446	0.969

Table 3: LightGBM Performance on Column-wise Inflation Adjusted Dataset

For our LightGBM models we also generated split and gain graphs to visualize what features the models were using to learn. The plots are shown below.

Finally, training a neural network only on cluster 1 yielded an RMSE of 20933.91 and this method was discontinued.

To improve modeling results across all clusters, we adopted Shawn's refactored pipeline, which introduced several key improvements to model training and evaluation. One major fix was ensuring that feature encoding and transformations were performed within the training loop for each cluster. This eliminated data leakage and prevented contamination across cluster subsets, which had previously distorted training outcomes.

We also addressed numerical inconsistencies by extracting X_numerical directly from X, rather than the original df, guaranteeing that the correct features were consistently used after encoding. To further stabilize the target variable, we applied a log-transformation to the target 2024_price, which proved crucial in reducing the impact of extreme price values—particularly beneficial in clusters with high variance, such as cluster 1.

Additionally, we corrected the RMSE computation by manually applying the square root of mean_squared_error, resolving prior inconsistencies that affected model evaluation. With these enhancements, the column-wise inflation-adjusted dataset showed significantly better and more stable results across all clusters.

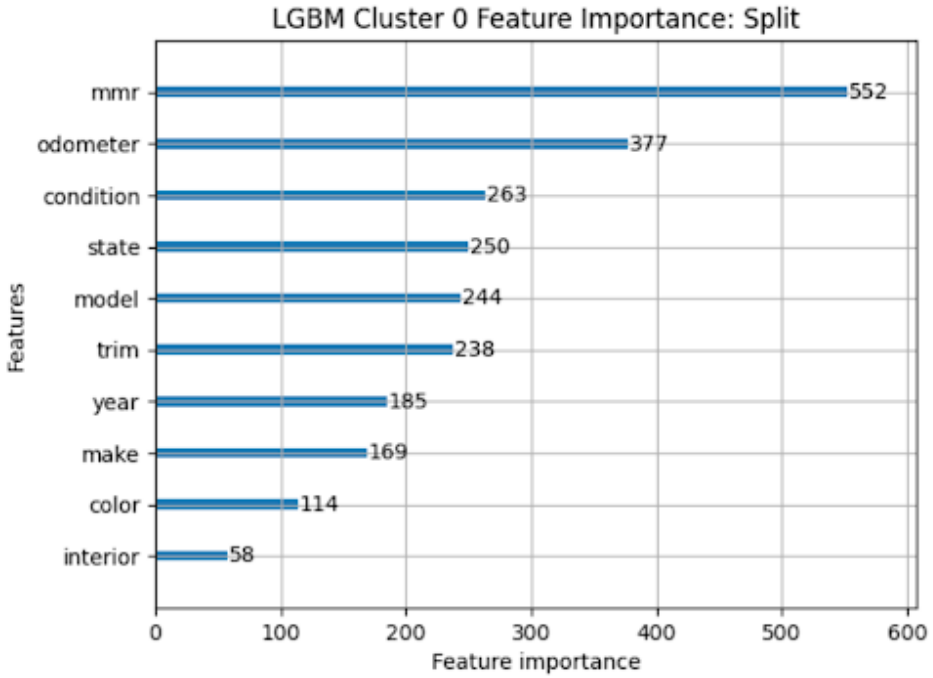

Figure 3: LGBM Cluster 0 Feature Importance: Split

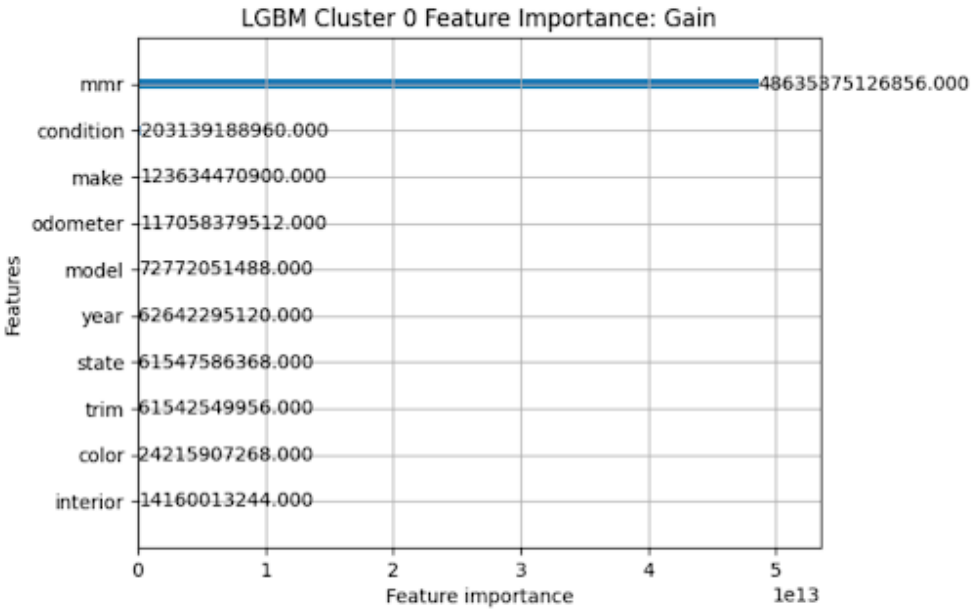

Figure 4: LGBM Cluster 0 Feature Importance: Gain

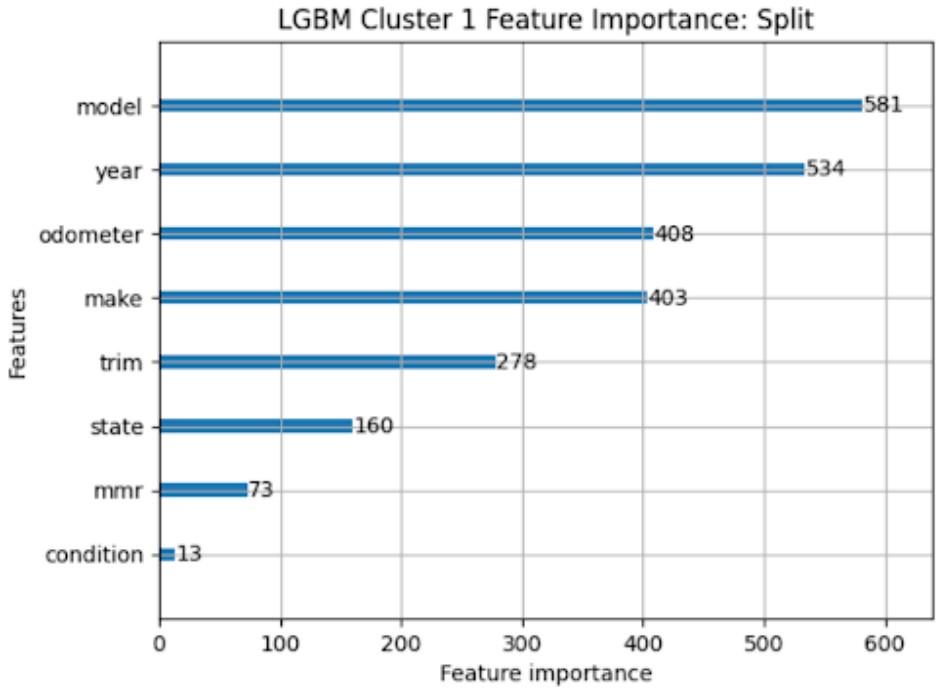

Figure 5: LGBM Cluster 1 Feature Importance: Split

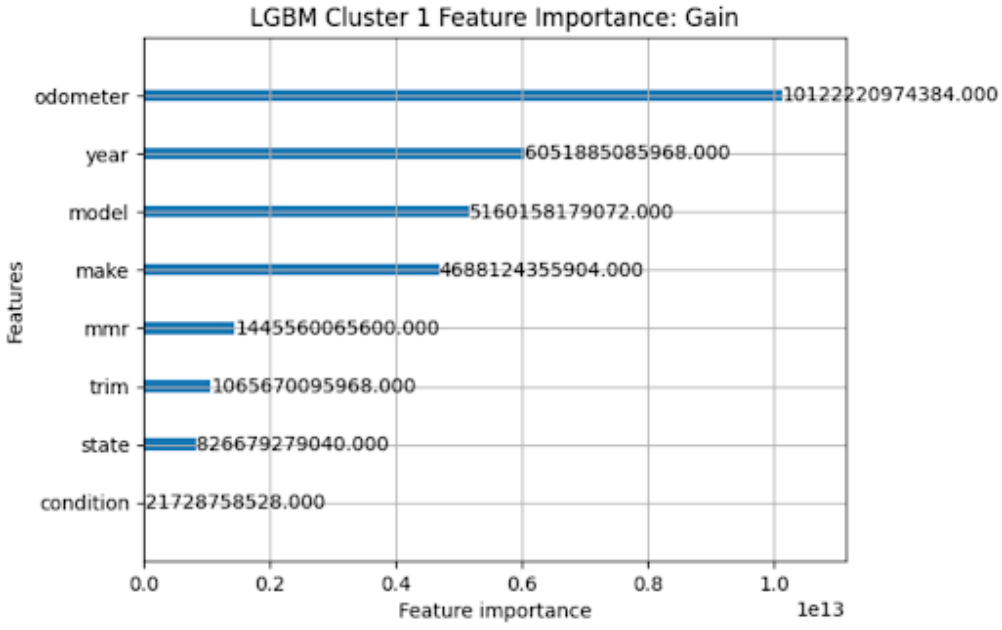

Figure 6: LGBM Cluster 1 Feature Importance: Gain

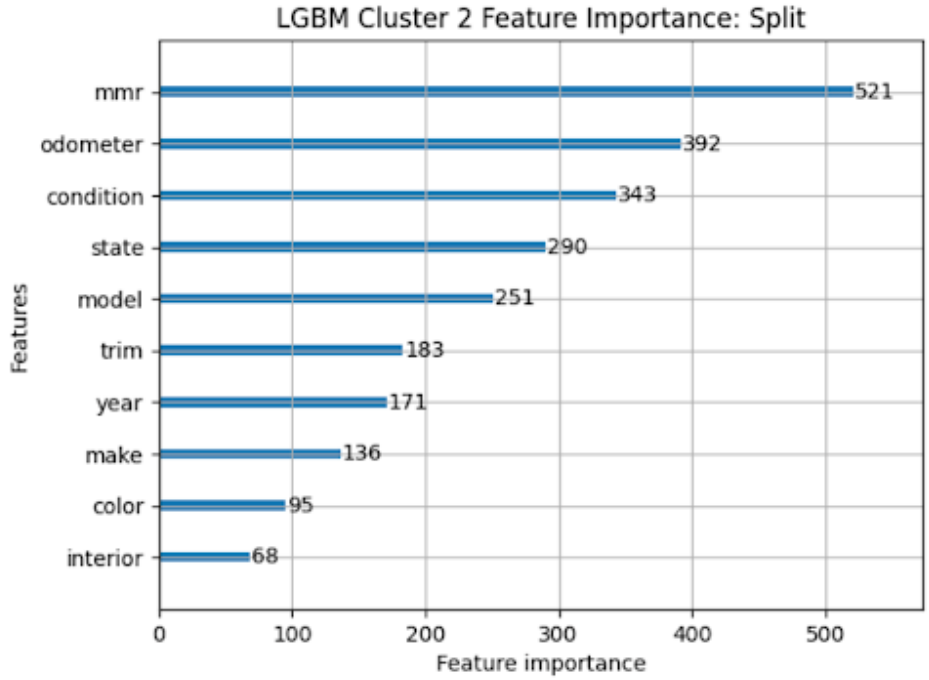

Figure 7: LGBM Cluster 2 Feature Importance: Split

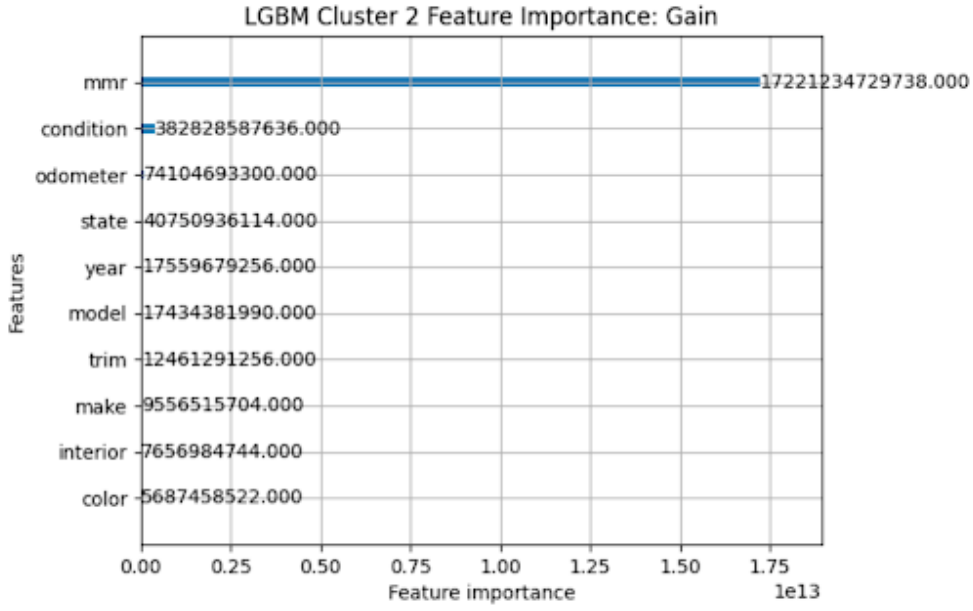

Figure 8: LGBM Cluster 2 Feature Importance: Gain

This improved setup achieved strong performance, particularly in Cluster 0 (RMSE: 1076, R^2: 0.9495) and Cluster 2 (RMSE: 2855, R^2: 0.9152), reflecting high predictive accuracy. While Cluster 1 remained more challenging due to its underlying heterogeneity, the revised pipeline still improved its performance considerably, yielding an RMSE of 6426 and R^2 of 0.6489, compared to earlier, much lower baselines. These outcomes confirmed that column-wise inflation adjustments and proper per-cluster processing were essential to boosting model generalizability and reducing systemic prediction error.

5.3 Row Wise Inflation Adjustment

To account for inflation, we normalized the 2024 sale price for every combination of year, make, model, and trim. We then grouped the dataset into two groups based on KMeans considering only numerical features. We used XGBoost and LightGBM models subsequent to clustering. These models were compared on RMSE and R^2 and hyperparameter tuning was performed to enhance their performance.

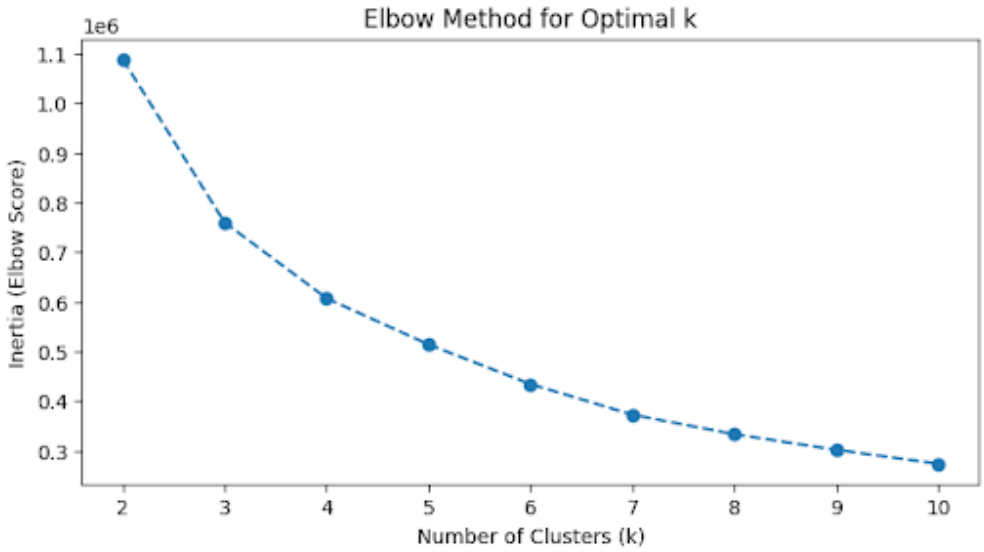

Figure 9: Elbow Method for Optimal k (Row-wise Approach)

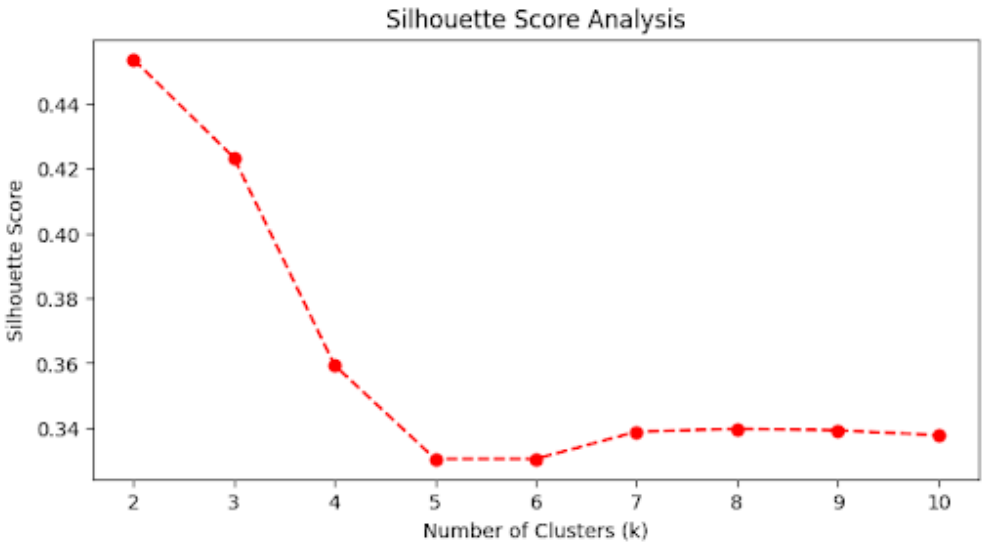

Figure 10: Silhouette Score Analysis

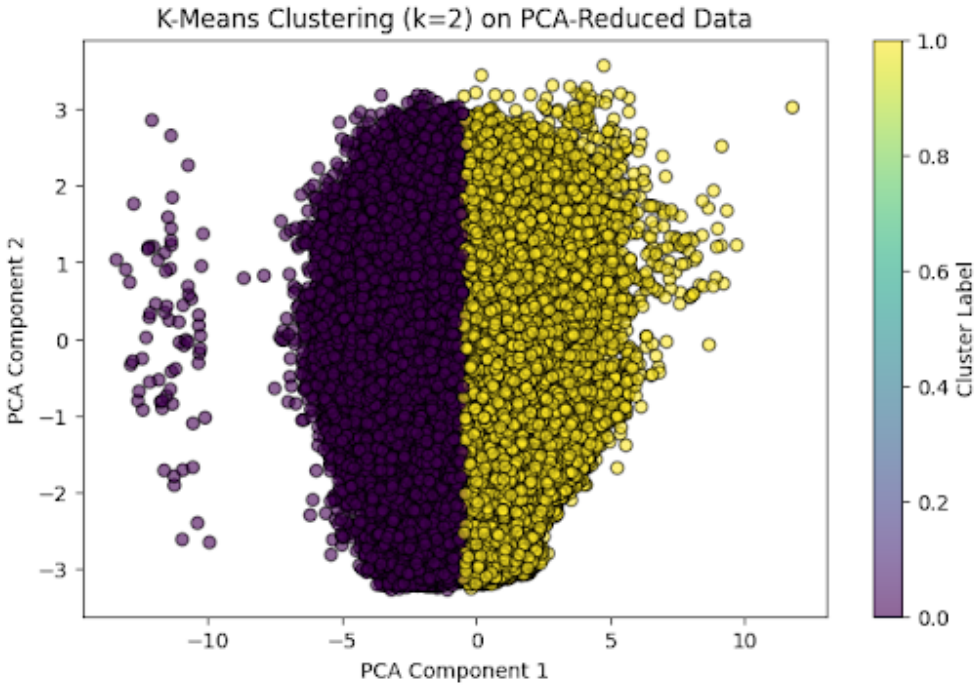

Figure 11: K-Means Clustering (k=2) on PCA-Reduced Data

Test	RMSE	R^2
Cluster: 0 XGBoost	2525.35	.912
Cluster: 0 LightGBM	2520.24	.965
Cluster: 1 XGBoost	2084.13	.872
Cluster: 1 LightGBM	2587.11	.921
Cluster: 2 XGBoost	2084.13	.932
Cluster: 2 LightGBM	2087.20	.933

Table 4: Model Performance on Row-wise Inflation Adjusted Dataset

6 Discussion

Our experimental results demonstrate that the column-wise inflation adjustment provided the most consistent performance improvement across all clusters. The default LightGBM model performed exceptionally well for clusters 0 and 2, while cluster 1 required additional feature engineering and hyperparameter tuning to achieve acceptable results.

Cluster heterogeneity presents a significant challenge in the automotive pricing domain. Vehicles within certain segments (like cluster 1) exhibited more varied price-feature relationships, necessitating more complex modeling approaches. This reflects the real-world challenges in the automotive market, where luxury, specialty, and rare vehicles often follow different pricing patterns than mainstream models.

The effectiveness of our clustering approach validates our hypothesis that segmenting the vehicle market improves predictive accuracy. By handling each segment with tailored

models, we achieved better overall performance than would be possible with a single model approach.

7 Conclusion and Future Work

After experimentation with testing our datasets and finding that the column-wise inflation-adjusted dataset best performed during the modeling process, we chose this variant as our baseline for the backend database powering our application. For each cluster, we saved and stored the model that performed best in our experiments. The resultant models were dumped using the joblib library for fast, low-footprint loading during inference.

For clusters 0 and 2, the best-performing models were LightGBM regressors with the default parameters (num_leaves=50, n_estimators=100, learning_rate=0.5, random_state=42). These models performed well out-of-the-box because there was comparatively clean separability in the feature space after clustering and benefited from the inflation adjustment that minimized temporal variance in price data.

Cluster 1, however, struggled at first. This cluster contained more diverse data and noisier feature-price relationships. To try to correct this, we introduced a second stage of optimization for cluster 1. We first introduced interaction features to capture more complex relationships—such as odometer per year, MMR to odometer ratio, and a multiplicative year-condition interaction. These were intended to bring nonlinear factors into the model's understanding, especially where age, use, and expected market value intersected with one another in intricate ways.

These models were compared based on RMSE as our baseline measure, which is both interpretable and aligned with the practical goal of minimizing price prediction error. Final RMSE values confirmed the efficacy of our improvements:

- Cluster 0 (LGBM default): RMSE $\sim$ 2503.47, $R^2 \sim 0.9612$
- Cluster 1 (LGBM tuned + feature engineered): RMSE $\sim$ 5595.76, $R^2 \sim 0.7072$
- Cluster 2 (LGBM default): RMSE $\sim$ 2124.32, $R^2 \sim 0.9717$

At deployment time, the system utilizes a learned KMeans model and normalized features to assign every new input vehicle to a cluster. The corresponding preprocessing pipeline—scalers and label encoders—are dynamically imported on a per-cluster basis, and predictions are made by using the best-performing LightGBM model for a given cluster. This architecture allows for modular model updates and ensures that the inflation-adjusted, cluster-optimal solution generalizes well across a wide range of vehicle profiles.

By combining inflation-aware preprocessing, unsupervised clustering, tailor-made feature engineering, and per-cluster modeling techniques, we could significantly reduce prediction error and develop a backend capable of efficiently backing a real-time car price forecast web application.

For future work, we recommend:

- Exploring more complex and efficient clustering methods to improve cluster homogeneity

- Implementing specialized neural networks for each cluster once sufficient data segmentation is achieved
- Expanding the dataset to include a broader range of years and more diverse vehicle types
- Adding functionality to retrain models periodically using user-submitted data on actual purchases
- Incorporating additional external factors such as regional economic indicators and fuel price trends

These enhancements would allow the models to adapt over time to new trends and would lead to more realistic results for our users.

References

References

[1] Kaggle, *Vehicle Sales Data*, https://www.kaggle.com/datasets/syedanwarafridi/vehicle-sales-data

[2] Kaggle, *US used car sales data*, https://www.kaggle.com/datasets/tsaustin/us-used-car-sales-data

[3] Ke, G., et al., *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*, NIPS, 2017.

[4] MacQueen, J., *Some methods for classification and analysis of multivariate observations*, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[5] Gegic, Enis, et al., *Car Price Prediction Using Machine Learning Techniques*, International Conference on Information Technology, 2019.

[6] Pillai, Aravind Sasidharan, *A Deep Learning Approach for Used Car Price Prediction*, Journal of Automotive Engineering, 2021.

[7] Yang, Richard R., Steven Chen, and Edward Chou, *Predicting Used Car Prices with Deep Learning*, Stanford Research, 2021.

[8] Varshitha, Janke, K. Jahnavi, and C. Lakshmi, *Prediction of Used Car Prices Using Artificial Neural Networks and Machine Learning*, IEEE Xplore, 2022.