Used Car Price Prediction

Qianhua Zhou (qz33)
Jerry Liang (zhirong5)
Sirui Cao (siruic6)

2024-12-12

Contents

Project Description and Abstract

This project focuses on predicting used car prices by leveraging advanced machine learning techniques and a meticulous data preprocessing pipeline. The dataset was extensively cleaned to address missing values, categorize key variables such as mileage, brand, and year, and extract detailed features from complex fields like engine specifications and transmission types. These steps ensured a robust foundation for subsequent analysis.

Unsupervised learning techniques played a critical role in enhancing data understanding and improving feature space organization. Principal Component Analysis (PCA) was applied to reduce dimensionality, simplifying the dataset into a manageable feature set while preserving essential information. This dimensionality reduction not only improved clustering performance but also facilitated insightful visualizations for exploring inherent data patterns. Clustering algorithms, including the Self-Organizing Map (SOM), were further employed to segment the dataset based on shared characteristics, uncovering structural relationships within the data and their impact on price predictions. By combining PCA and clustering, the project demonstrated the power of unsupervised learning to enhance data interpretability and predictive performance.

Unlike the models reviewed in the Literature Review, which focused on basic preprocessing and standard model tuning, this project integrated advanced feature engineering techniques. These included systematically extracting and categorizing complex variables, such as engine specifications and transmission details, which were often overlooked in prior studies. This level of detail provided a competitive advantage, improving the models' accuracy and interpretability when working with high-dimensional and mixed-type data.

The predictive component of the project centered on designing and fine-tuning five machine learning models:

- Elastic Net: A linear regression model combining L1 and L2 penalties to address multicollinearity and perform feature selection, ensuring robust predictions.
- K-Nearest Neighbors (KNN): A non-parametric approach using feature similarity (measured by Euclidean distance) to predict prices based on the average of neighboring observations.
- Gradient Boosting Machine (GBM): An ensemble learning method combining weak learners (decision trees) to create a strong predictive model, optimized through iterative boosting techniques.
- Random Forest: A powerful ensemble method that constructs multiple decision trees, averaging their predictions to minimize overfitting. This model achieved the highest performance, with a normalized Mean Squared Error (MSE) of 0.2528, emphasizing the importance of features such as model year, mileage, and brand.
- Neural Network: A deep learning model designed to capture complex non-linear relationships in the dataset, providing competitive predictions after careful architecture tuning.

All models underwent rigorous hyperparameter optimization using grid search and cross-validation to ensure optimal performance. The results underscored the importance of combining rigorous preprocessing, feature engineering, and advanced machine learning techniques to address real-world predictive challenges effectively.

Key findings from this study emphasize the value of rigorous data preprocessing and feature engineering in handling high-dimensional and mixed-type datasets. The integration of unsupervised techniques, such as PCA and clustering, with predictive models improved both accuracy and interpretability. This approach demonstrates the potential of combining dimensionality reduction with machine learning in predictive modeling tasks. Overall, this project showcases a robust framework for data-driven problem-solving, offering valuable insights for research and industry applications.prediction, offering valuable insights for both research and industry applications.

Statement of AI usage: We use ChatGPT to generate ideas for potential model selections, then write the code part by ourselves. And after we complete the draft version, we refine the description with GPT's help.

Literature Review

Paper1: Samruddhi, K., & Kumar, R. A. (2020). Used car price prediction using K-nearest neighbor based model. Int. J. Innov. Res. Appl. Sci. Eng.(IJIRASE), 4(3), 2020-686.

Methodology K-Nearest Neighbor (KNN) Regression

- Purpose: Predict prices based on feature similarity measured by **Euclidean distance**.
- Dataset: Use dataset from Kaggle, featuring variables such as mileage, engine type, fuel type, etc.
- Data Preprocessing: Removed non-numerical parts from numerical features. Converted categorical variables (e.g., fuel type, transmission) into numerical representations. Separated the target variable (`Price`) from the feature set.
- Model Training: Implemented KNN to predict prices by averaging the values of the k nearest neighbors. Evaluated the model for k values ranging from 2 to 10. Performed hyperparameter tuning to identify the optimal k value.
- **Cross-Validation**: Applied k-fold cross-validation (5-fold and 10-fold) to assess model generalizability and avoid overfitting.
- Evaluation Metrics: Root Mean Squared Error (RMSE). Mean Absolute Error (MAE).

Findings

Optimal Performance: Best results achieved with **k=4**, delivering: Accuracy: 85%. RMSE: 4.01. MAE: 2.01.

Cross-Validation Results: 10-fold cross-validation yielded an accuracy of **82%**.

Comparison with Linear Regression: KNN significantly outperformed linear regression, which had an accuracy of **71%**.

Paper2: Pal, N., Arora, P., Kohli, P., Sundararaman, D., & Palakurthy, S. S. (2019). How much is my car worth? A methodology for predicting used cars' prices using random forest. In Advances in Information and Communication Networks: Proceedings of the 2018 Future of Information and Communication Conference (FICC), Vol. 1 (pp. 413-422). Springer International Publishing.

Methodology

Random Forest Regression

Purpose: Predict used car prices by leveraging the ensemble learning technique of Random Forest regression.

- Dataset: Utilized the Kaggle Used Car Dataset containing over 370,000 records with attributes like price, mileage, brand, vehicle type, etc.
- The data preprocessing phase involved removing irrelevant columns and filtering out unrealistic entries, such as cars manufactured before 1863. Boolean fields were converted to numeric values, and missing data was systematically handled. From the cleaned dataset, ten critical features—price, vehicleType, age, powerPS, kilometer, fuelType, brand, and others—were selected to build a robust predictive model.
- For model training, the dataset was split into training (70%), testing (20%), and validation (10%) subsets to ensure reliable evaluation. Hyperparameters were optimized using a Grid Search Algorithm, focusing on the number of decision trees (500) and the maximum features considered at each split. The Random Forest model was applied for its strength in minimizing overfitting by averaging predictions across multiple decision trees, yielding reliable and accurate results.
- Evaluation Metrics: Coefficient of Determination (**R^2**). Accuracy scores for training and testing data.

Findings

- Optimal Performance: Achieved a training accuracy of **95.82%** and testing accuracy of **83.63%**. The model effectively captured critical relationships between features and the target variable, producing reliable predictions.
- Feature Importance: The analysis identified the most impactful features influencing price prediction: price (target variable), kilometer (mileage), brand, and vehicleType. These features played a crucial role in driving the model's accuracy and interpretability, underscoring their significance in the predictive framework.
- Comparison: Demonstrated superior accuracy compared to linear regression and other simpler models, addressing overfitting through ensemble averaging.
- Future Directions: Propose exploring advanced techniques like fuzzy logic and genetic algorithms for further improvement. Aim to develop an interactive system with a recommendation engine for predicting prices based on user-input features.

This methodology highlights the effectiveness of Random Forest regression in solving complex prediction tasks while maintaining generalizability and interpretability.

Data Processing and Summary Statistics

Brand & Model The original dataset does not contain missing values for the "Brand" variable. Since "Brand" consists of 57 distinct values with a relatively sparse distribution, we grouped these brands into broader categories: "Cheap," "Mainstream," "Luxury," or "Other" based on well-known market associations. For this task, we utilized the `dplyr::case_when()` for conditional assignment and `stringr::str_detect()` for efficient text matching.

Model Year The `model_year` variable has no missing values, with a range from 1974 to 2024. While most values fall between 2008 and 2023, the distribution is relatively sparse. To address this, we grouped model_year into four categories for easier analysis: "New"(2020~), "Recent"(2015~2020), "Old"(2010~2015), or "Very Old"(~2010). This grouping ensures a more balanced distribution of observations across categories. We use `dplyr::case_when()` for grouping.

Milage The `mileage` variable contains no missing values. We first removed non-numeric characters "mi." using function `gsub()` and categorized mileage into tiers. After converting to numeric type, we found the range vary from 0 to more than 150000, and the distribution is sparse. So we use `dplyr::case_when()` grouped milage into four categories: "Low Mileage"(~25000), "Moderate Mileage"(25000~75000), "High Mileage"(75000~150000), or "Very High Mileage"(150000~).

Engine The `engine` in the original dataset contained a mix of information, including details about horsepower, engine size, number of cylinders, fuel type, and cylinder arrangement. We systematically extracted and processed this information to create the following new variables:

- fuel_type: Extracted using case_when and str_detect to identify whether the engine was powered by gasoline, hybrid, electric, or diesel, with a fallback category of "Other."
- horsepower: Extracted numeric values corresponding to horsepower from patterns like "X.XHP" using str_extract and converted them into numeric format after removing the "HP" suffix.

- engine_size: Extracted engine size in liters from patterns like “X.XL” using str_extract and converted them to numeric format after removing the “L” suffix.
- cylinders: Extracted the number of cylinders from phrases like “X Cylinder” using str_extract and converted the result to numeric.
- cylinder_arrangement: Classified cylinder arrangements into categories such as “V,” “Flat,” and “Straight” using case_when and str_detect, with a fallback “N” for undefined arrangements.

After extracting, there are around 800 missing entries out of 4009, and we handle the missing values by replacing them with the column’s mean.

Transmission The original dataset’s transmission variable contains diverse formats, including speed counts (e.g., “6-Speed A/T”), transmission types (e.g., “Automatic”), and ambiguous entries (e.g., “F” or “A/T”). We extracts the transmission type from the transmission column in the data dataframe and adds a new column called transmission_type. Using conditional logic with `dplyr::case_when()`, we categorizes the transmission as “Automatic” if it contains keywords like “A/T”, “Automatic”, or “CVT”, or as “Manual” if it contains “M/T” or “Manual”. For all other values, we assigns “Other”.

Ext_col There is no missing values for `ext_col`. The original dataset’s ext_col variable contains diverse exterior colors and the distribution is sparse. So we use `dplyr::case_when()` categorize it into five categories:

- Dark: Includes shades like black, brown, or dark-related keywords.
- Light: Includes white, silver, grey, and light-related keywords.
- Vivid: Encompasses bright colors like red, blue, green, and vibrant-related terms.
- Unknown: Covers missing or uninformative values.
- Other: Any entry that doesn’t fit the above categories.

Int_col There is no missing values for `ixt_col`. The original dataset’s int_col variable contains diverse exterior colors and the distribution is sparse. So we use `dplyr::case_when()` categorize it into five categories, similar to the process for `ext_col`.

Accident The original dataset’s accident variable contains values indicating accident history, such as “None reported,” “At least 1 accident or damage reported,” and missing entries(NA).The amount of missing entries is relative low(113 out of 4009), so we assume all the “NA” is equibalent to “None reported”. So we recode it as: “None reported”/“NA” -> 0; “At least 1 accident or damage reported” -> 1

Clean Title The code summarizes the clean_title column, counting “Yes,” “No,” and missing values using table(). It then recodes “Yes” to 1, “No” to 0, and replaces NA values with 0, ensuring the column is binary and complete for further analysis or modeling.

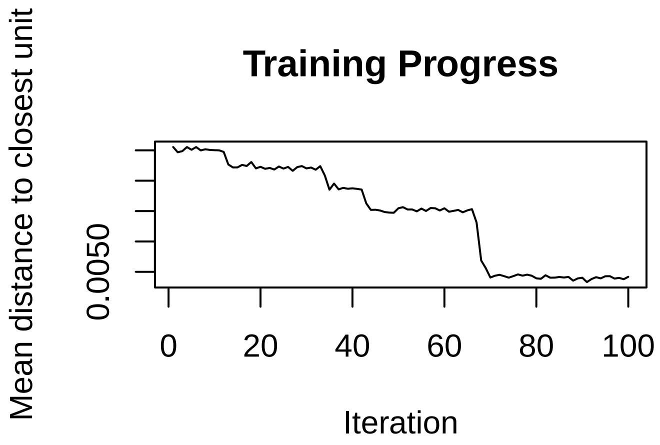
Price There is no missing values for `price`. The original dataset’s `price` variable contains diverse exterior colors and the distribution is sparse. So we use `dplyr::case_when()` categorize it into five categories: “< $12,000”, “$12,000 - $21,999”, “$22,000 - $31,999”, “$32,000 - $49,999”, “$50,000 - $79,999”, “> $80,000”.

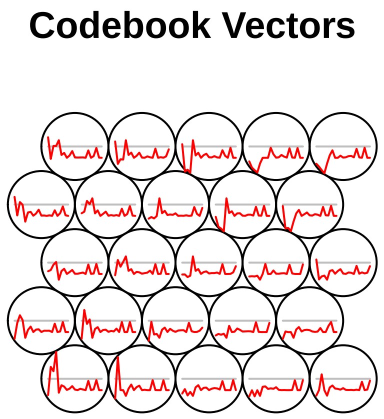
Table of summary statistics:

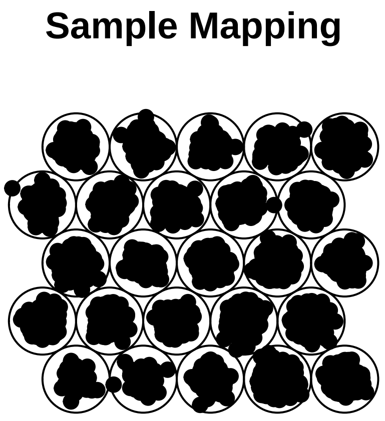
```
##
## Descriptive Statistics
## ========================================================
## Statistic     N       Mean      St. Dev.    Min       Max
## --------------------------------------------------------
## milage      4,009 64,717.550 52,296.600  100     405,000
## horsepower  4,009  332.284     109.199   70.000 1,020.000
## engine_size 4,009   3.711       1.352    0.650    8.400
## cylinders   4,009   6.258       1.466    3.000   12.000
## accident    4,009   0.246       0.431      0        1
## clean_title 4,009   0.851       0.356      0        1
## price       4,009 44,553.190 78,710.640 2,000  2,954,083
## --------------------------------------------------------
```

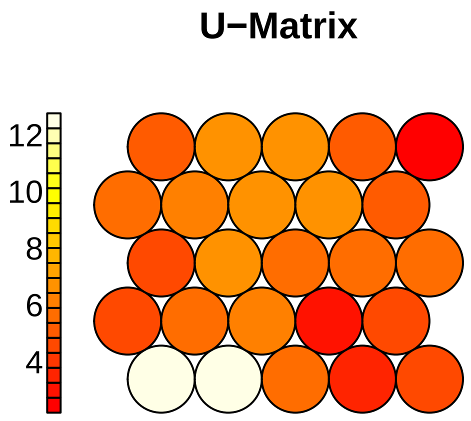
Refer the supplementary section for histogram plots, boxplots, correlation heatmap, frequency table, etc.

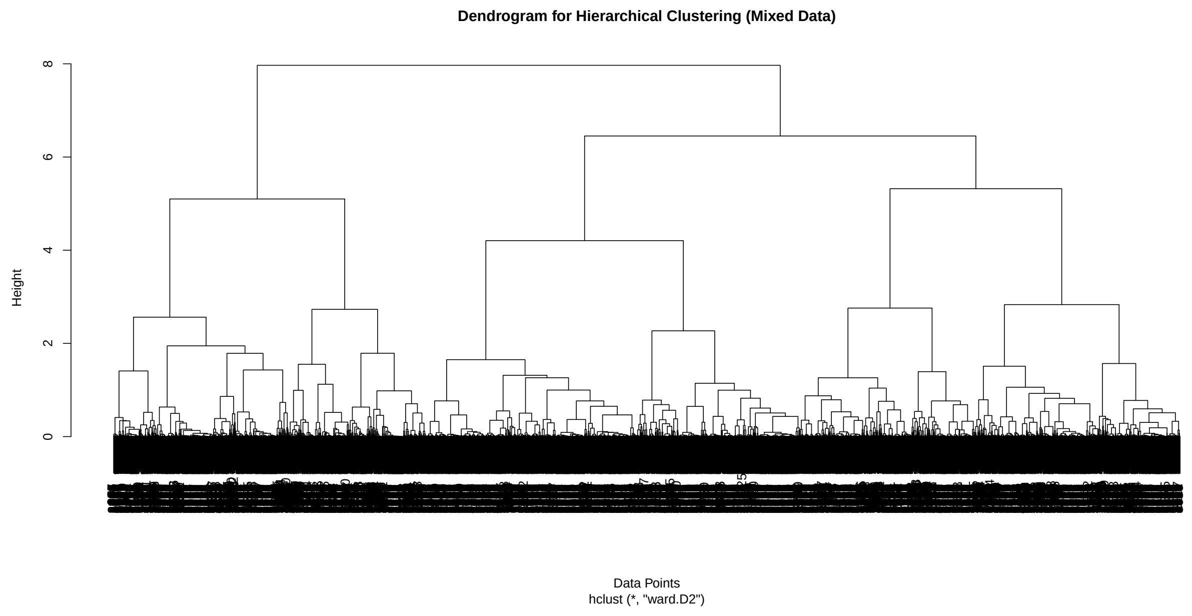
Unsupervised Learning











```
## [1] "Cluster sizes: "

##
##    1    2    3
## 1177 1429 1403
```

Among the three clustering methods evaluated, K-Means clustering with PCA emerged as the best, achieving a silhouette score of 0.5447 compared to 0.2405 for hierarchical clustering and 0.1746 for Self-Organizing Maps (SOM). SOM, which maps high-dimensional data onto a low-dimensional grid, struggled with cluster separability due to its reliance on topological preservation, resulting in poor clustering quality. Hierarchical clustering, utilizing Gower distance to handle mixed data types, provided slightly better results but remained limited by its sensitivity to noise and computational intensity. K-Means with PCA outperformed the others by leveraging dimensionality reduction to simplify data structure, mitigating the challenges posed by high-dimensional features and dummy variables for categorical data. PCA effectively retained the most informative variance while reducing noise and redundancy, enabling K-Means to form compact, well-separated clusters. This combination of PCA and K-Means proves particularly effective for large, mixed-type datasets, balancing computational efficiency and clustering accuracy.

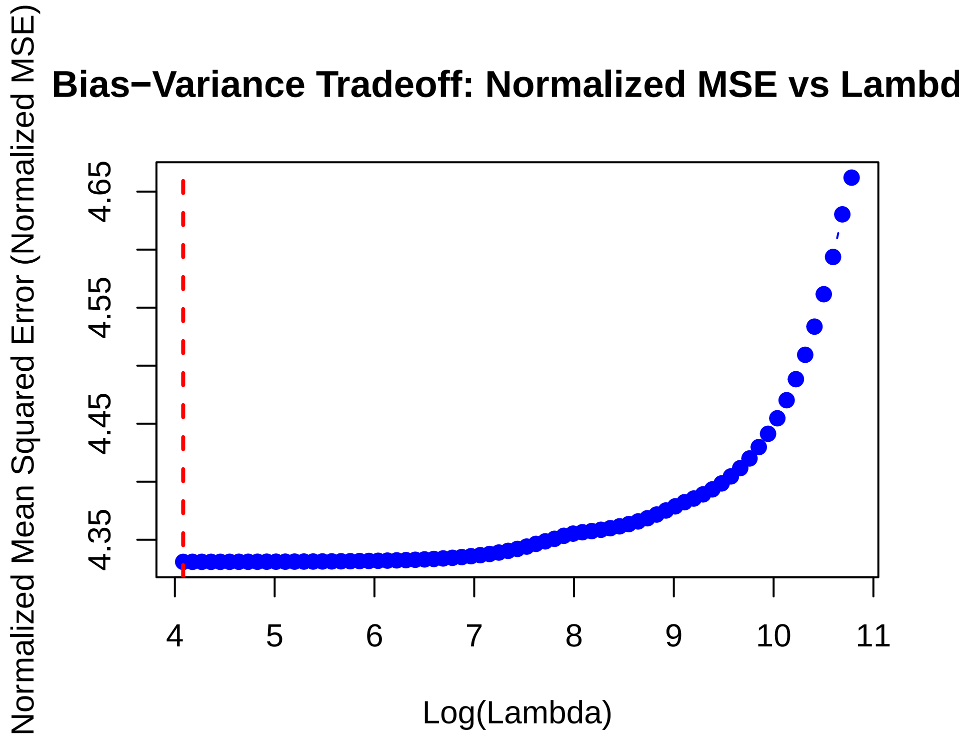
Categorical data was handled by transforming it into dummy variables, where each category was represented as a binary column indicating presence or absence. While this increases the dimensionality of the dataset, it allows clustering methods to treat categorical variables as numerical features. Hierarchical clustering addressed this using Gower distance, which combines numerical and categorical similarity metrics, ensuring mixed data types were treated appropriately. For K-Means, PCA effectively reduced the high-dimensional space resulting from dummy encoding, integrating categorical and numerical features into a low-dimensional space that captured the most significant variance in the data, enabling better clustering performance.

The clustering and dimensionality reduction results previously obtained can significantly enhance supervised methods by simplifying the data structure and introducing high-level features, which can be directly used in some methods that only applies for low dimensional data. Principal Component Analysis (PCA) reduced the dimensionality of the data while preserving most of its variance. For instance, the resulting principal components can be directly used as predictors in regression models or classification tasks, ensuring that only the most informative aspects of the data are included, which simplifies computation and improves model interpretability. Clustering added an extra layer of information by grouping data points based on similarity, effectively creating new categorical variables that represent latent structures within the data. These cluster labels or their distances from centroids can be incorporated as predictors, enriching supervised models with information about subgroup-level behaviors or characteristics that may influence the target variable.

Refer the supplementary section for other related plots.

Prediction Models

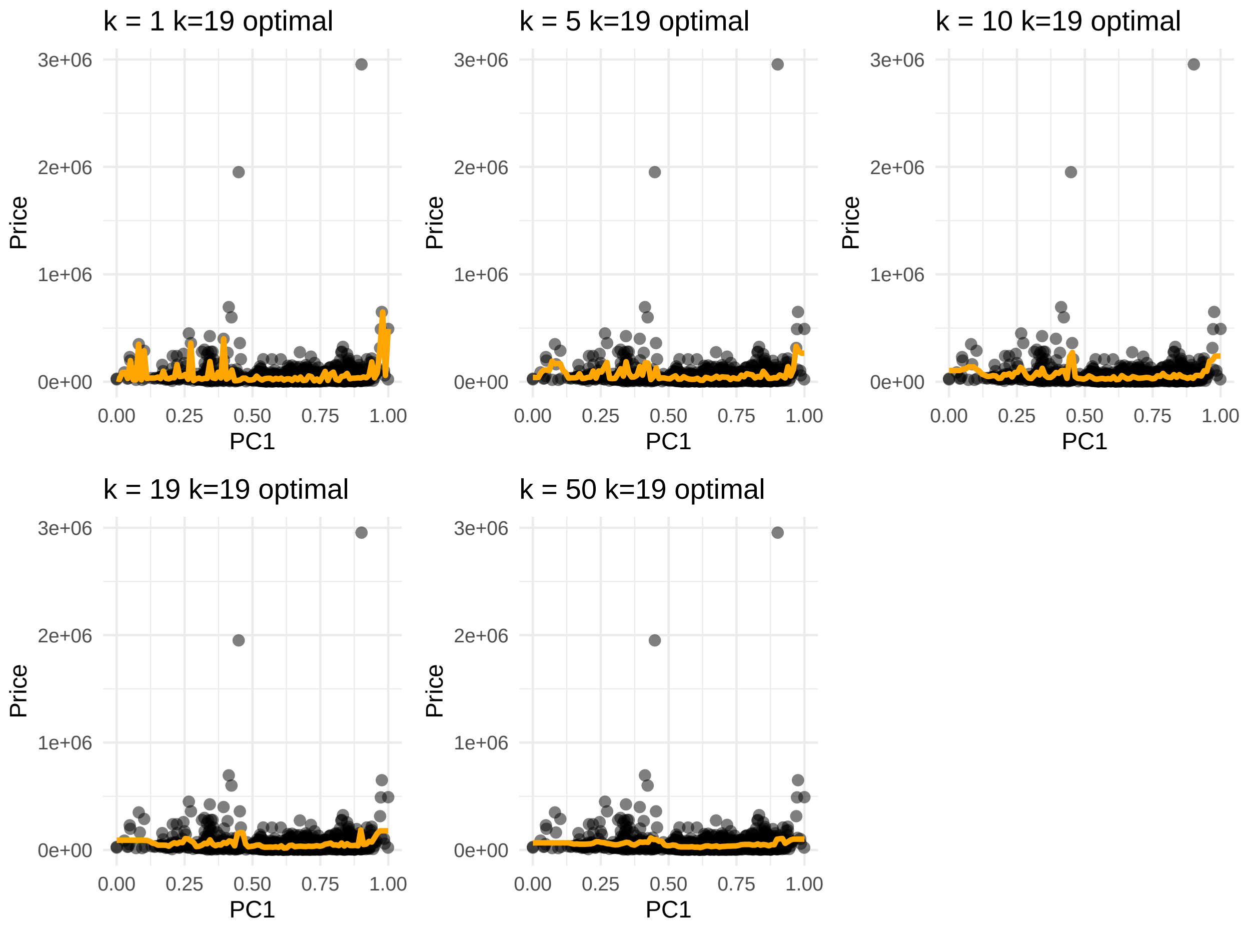
Model 1: elastic net, where we used PCAed data and cluster directly



In the Elastic Net regression example, PCA was used to extract the top two principal components, reducing the feature space while retaining the most critical information. These components were combined with cluster labels derived from the unsupervised learning process, creating a dataset where predictors are both low-dimensional and enriched with grouping information. Elastic Net, which balances L1 and L2 regularization, benefits from this reduced complexity and the removal of redundant information, leading to a better generalization. By including cluster labels, the model gains insight into potential segment-level differences, improving its ability to predict the target variable. The combination of PCA and clustering directly supports Elastic Net's linear assumptions and enhances its performance on high-dimensional data.

The Elastic Net model results demonstrate a well-balanced approach to predictive modeling, leveraging both L1 and L2 regularization with an alpha value of 0.5. This balance ensures feature selection through sparsity while maintaining stability in the presence of multicollinearity. The bias-variance tradeoff plot highlights the optimal lambda value of approximately 59.34, which minimizes the normalized mean squared error (MSE), achieving the best tradeoff between model complexity and predictive performance. At lower lambda values, weaker regularization risks overfitting, while higher values result in underfitting due to overly constrained coefficients. The scatter plot of predicted vs. actual prices reveals that the model performs well for most data points, especially in the lower price ranges, as predictions align closely with the actual values. However, for higher-priced items, the predictions exhibit significant scatter, suggesting underprediction and reduced accuracy for outliers or extreme values. This indicates a potential need for additional feature engineering or targeted data augmentation to better capture variability in high-value cases. The model's use of dimensionality-reduced features (via PCA) combined with cluster information likely contributed to its ability to generalize for most cases.

Model 2: KNN, where we used PCAed data and cluster directly

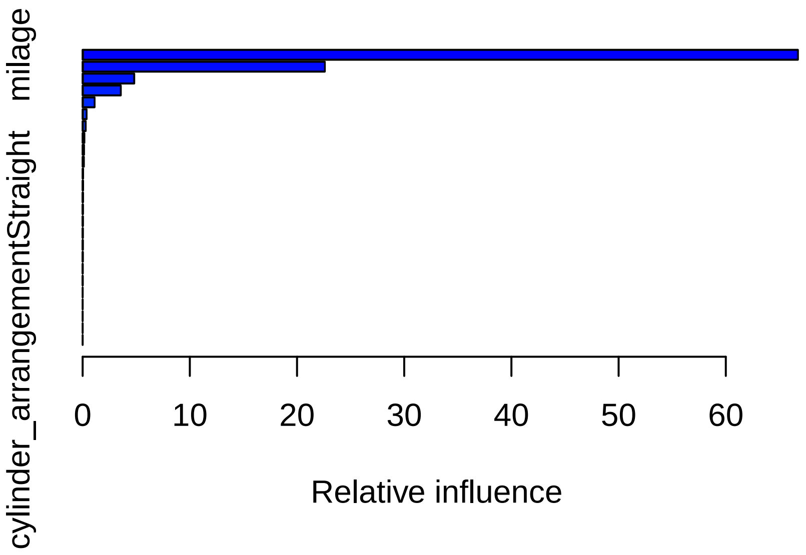


Similarly, in the k-Nearest Neighbors (k-NN) model, PCA helped overcome the curse of dimensionality by projecting the data onto its first two principal components, making it suitable for distance-based methods. Clustering results were also incorporated as an additional feature, providing context about group-level similarities that k-NN could leverage. The predictors were normalized to ensure that all features contributed equally to the distance calculations. Cross-validation was used to tune the hyperparameter k, ensuring the optimal balance between bias and variance. The integration of unsupervised methods like PCA and clustering not only improved k-NN's predictive power but also ensured the model's robustness and interpretability across varying data complexities.

The k-Nearest Neighbors (k-NN) regression results indicate that the best value for k, determined through cross-validation, is 19, which minimizes the normalized mean squared error (MSE) to approximately 2.95. This value of k provides the optimal balance between bias and variance.

Considering the complexity of the nature of this problem, linear method and low dimensional method like elastic net and knn might not perform well, however we still perform it as a reference or demonstration of skill.

Model 3: GBM, where we use both original data, pc1, 2, and cluster



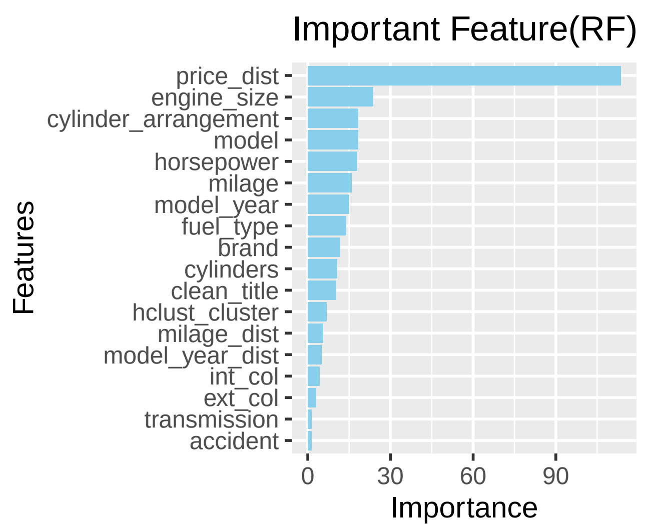
The Gradient Boosting Model (GBM) was trained using a dataset that integrated PCA-transformed features, cluster labels from k-means clustering, and the original scaled and dummy-encoded features to predict the logarithmic transformation of price (`log_price`). After splitting the data into training and test sets, the model determined the optimal number of trees to be 700 based on 8-fold cross-validation. The model achieved an r-squared of 0.75391 on the training data and 0.751 on the test data, indicating that it explains most of the variance in the logarithmic price. The normalized mean squared error (NMSE) on the test set was 3.4917, reflecting good predictive accuracy. The variable importance analysis identified mileage, horsepower, and engine size as the most influential predictors, aligning with expectations regarding their impact on vehicle valuation. This methodology effectively demonstrates the benefits of combining PCA, clustering, and GBM to handle complex, high-dimensional data and improve predictive performance. Consider the high dimensionality of orginal data and the output, GBM is the best model among all so far. (Refer the supplementary section for related plots.)

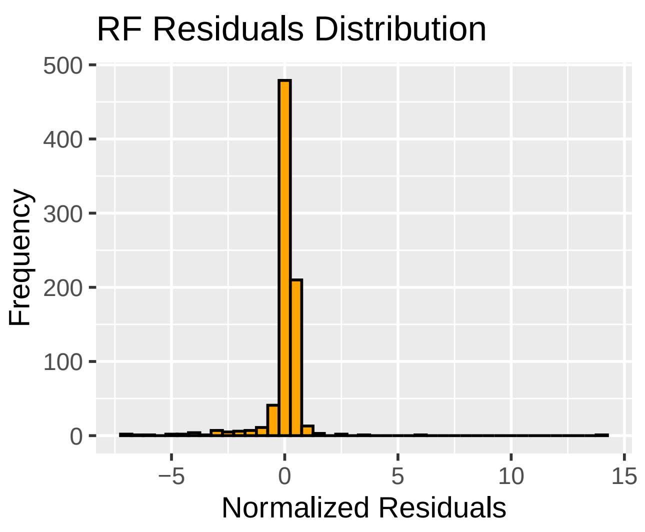
Model 4: Random forest, where we used features in original data directly

```
##   mtry
## 5   10
```

The Random Forest model, an ensemble learning method based on decision trees, was utilized for predicting car prices. This method constructs multiple decision trees during the training phase and combines their outputs to improve prediction accuracy. Random Forest is particularly effective in capturing non-linear relationships among features and is robust against outliers, making it suitable for this dataset.

Tuning Process: To optimize the performance of the model, hyperparameters such as the number of features considered for splitting (mtry) and the number of trees (ntree) were tuned. A grid search approach was implemented to explore values for mtry, specifically 2, 4, 6, 8, 10. Five-fold cross-validation was performed to select the best value of mtry that minimized error. The optimal mtry was determined to be 10, ensuring that the model could balance bias and variance effectively.





After tuning, the model was trained with 500 trees using the optimal mtry. The following evaluations were performed:

The model identified the most significant predictors of car prices, such as price_dist, engine_size, and horsepower. A bar plot of feature importance was created to provide visual insights into the relative contributions of these features.

The NMSE of the Random Forest model was calculated as 0.1884, demonstrating strong predictive performance and a substantial reduction in error compared to earlier iterations. An R-squared value of 0.7961 on the test data implies that nearly 79.61% of the variability in car prices is explained by the Random Forest model, indicating a high level of explanatory power. This performance suggests that the model effectively captures the relationships among features and provides reliable predictions.

The residuals histogram shows that the residuals are largely centered around zero, with minimal skewness, indicating that the model captured most of the variance in the data without introducing significant bias. This confirms that the model is well-calibrated and robust against errors.

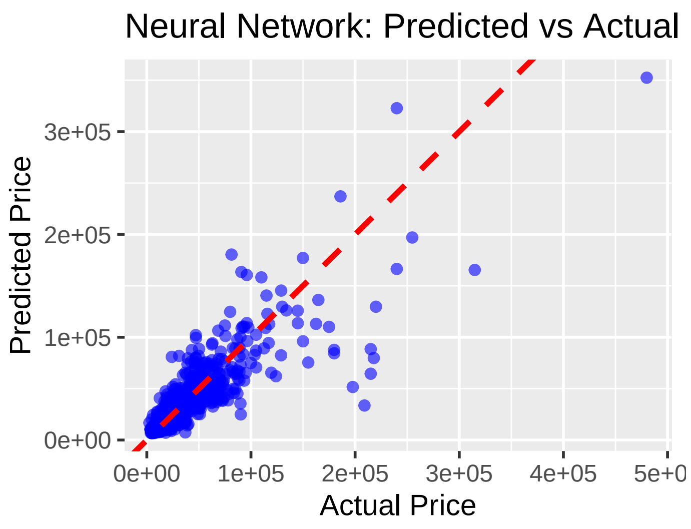
The Random Forest model performed exceptionally well in predicting car prices, with a high r-square, low NMSE, and interpretable results. The insights gained from feature importance analysis can guide strategic decisions, such as identifying which features are most valuable in influencing pricing. This model demonstrates its robustness and ability to handle non-linear relationships and outliers effectively, making it a strong candidate for deployment in predictive tasks related to car pricing.

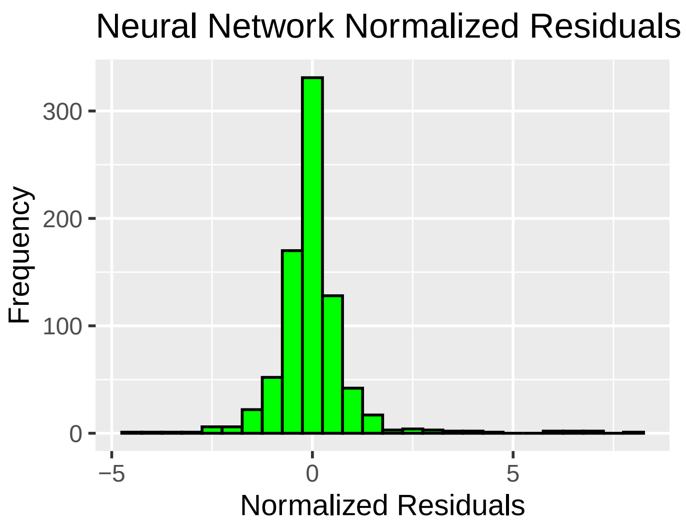
Model 5: Neural network, where we used PCAed data and cluster directly

```
##   size decay
## 9   10   0.5
```

In this model, we implemented a neural network to predict car prices. Neural networks are powerful tools for modeling complex, non-linear relationships between input features and the target variable. They are particularly useful for data with intricate interactions that may not be captured effectively by traditional models. We used features derived from principal component analysis (PCA) and clustering, combined with hyperparameter tuning, to optimize the performance of the model.

To optimize the performance of the neural network model, hyperparameters such as the number of hidden neurons (size) and the regularization parameter (decay) were tuned. A grid search approach was implemented to explore values for size (5, 10, 20, 30) and decay (0.001, 0.01, 0.1, 0.5, 1). Five-fold cross-validation was performed to evaluate each combination of hyperparameters and select the best configuration that minimized the Mean Squared Error (MSE). The optimal values identified were size = 10 and decay = 0.5, ensuring that the model could balance complexity and generalization effectively.





The neural network model was optimized and evaluated using the normalized mean squared error (NMSE). The best-tuned model achieved a Normalized MSE of 0.2852, which indicates a moderate deviation of predicted prices from actual prices, relative to the variance in the data.However, the r-square values reveal distinct trends: a relatively low r-square of 0.3384 on the training data suggests underfitting, while a higher r-square of 0.6925 on the test data indicates reasonable generalization to unseen data.

The scatter plot of predicted vs. actual prices highlights the model's performance: Most data points are clustered near the diagonal reference line (red dashed line), indicating acceptable predictive accuracy for many cases. Noticeable deviations from the diagonal are observed for higher price ranges, where the model struggles to capture complex patterns. This suggests that the neural network underfits high-priced vehicles, failing to capture their underlying variability.

The histogram of normalized residuals (errors scaled to unit variance) shows: Residuals are concentrated around 0, implying that predictions are generally close to actual prices for most instances. However, there is a slight right skew, with some residuals extending significantly beyond 5. This highlights that the model's predictions for certain high-priced vehicles deviate substantially from their true values.

The model performs reasonably well for mid-range prices, as evidenced by the cluster of points near the diagonal in the predicted vs. actual plot. However, the underperformance for higher price ranges suggests that the neural network may not fully capture the complex interactions between predictors for those cases.

Open-Ended Question

To estimate the original price of the cars in the dataset as if they were brand new, I would adopt a structured approach leveraging the dataset features and external data sources: #### 1. **Methodology** The first step involves utilizing the existing features in the dataset, such as brand, model year, mileage, fuel type, engine size, and transmission type, which serve as proxies for factors strongly correlated with the original price. I would incorporate the following steps:

- **Depreciation Modeling**: Analyze the relationship between current prices and depreciation factors like car age, mileage, and accident history. Depreciation typically follows a non-linear pattern, with the steepest decline in the first few years. A logarithmic or exponential depreciation model, informed by industry research, could approximate this relationship.

- **External Data Integration**: Enrich the dataset with historical data on new car prices for specific brands, models, and years, sourced from online databases, manufacturer reports, or market research studies. These external data points would act as benchmarks for training a predictive model.
- **Predictive Modeling**: Use a regression-based model to predict new car prices. A Random Forest regressor, which handles non-linearity and interactions effectively, is suitable for this task. The model would use features from the dataset as predictors, trained on the historical new price data.
- **Extrapolation**: For cars lacking external benchmarks, extrapolate using the model trained on comparable vehicles, considering depreciation trends and feature similarities. #### 2. **Challenges and Limitations**
- **Data Sparsity**: Older or less common models may lack sufficient historical price data, leading to less reliable predictions.
- **Feature Representation**: Factors influencing original prices, such as optional packages, trim levels, and market-specific adjustments, might not be included in the dataset, causing bias.
- **Extrapolation Risks**: The model might struggle to generalize accurately for new car prices based on data dominated by used cars.
- **External Data Integration**: Ensuring consistency and completeness when merging external datasets is complex and time-consuming. #### 3. **Prediction** Using a Random Forest regression model, I estimated the original prices for three selected cars from the dataset, comparing predictions to their actual new prices:

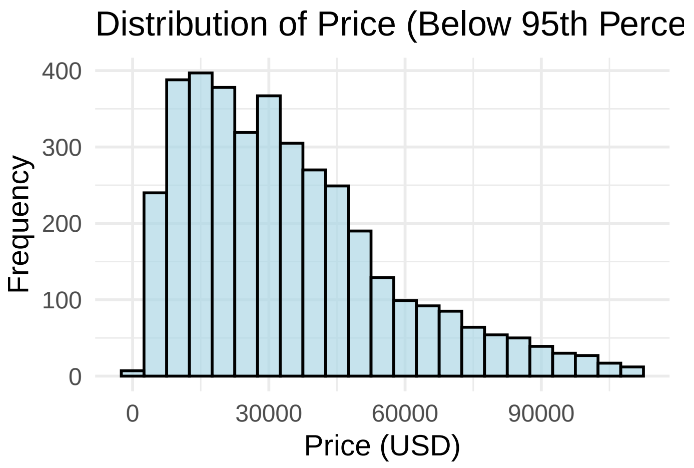
```
##          price_dist engine_size horsepower milage model_encoded
## 1 $12,000 - $21,999         3.5        354      0          1254
## 2         < $12,000         3.7        300      0          1758
## 3 $32,000 - $49,999         2.0        292      0          1472
##                              model predicted_price original_price discrepancy
## 1                 Q50 Hybrid Sport        45111.12          44400   -711.1172
## 2 Utility Police Interceptor Base        44181.33          29100 -15081.3310
## 3            S3 2.0T Premium Plus        52928.27          51325  -1603.2715
```

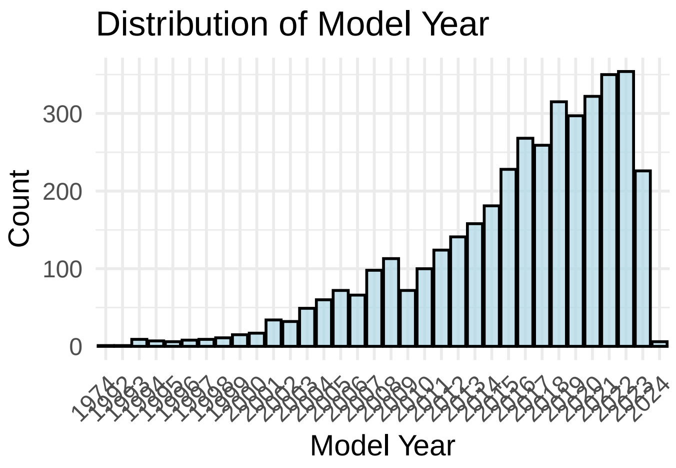
4. Discussion of Discrepancies The discrepancies between the predicted prices and the original Manufacturer's Suggested Retail Price (MSRP) for the selected cars highlight several challenges in modeling car prices. In this case, all predicted prices were higher than the actual new car prices. This overestimation can be attributed to the following factors:

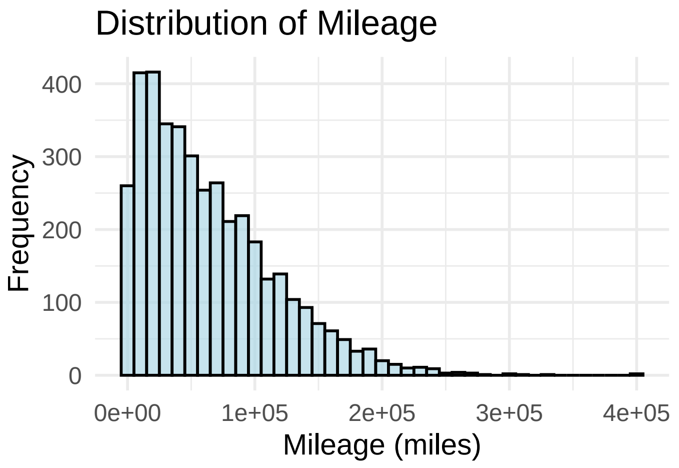
- **Feature Representation**: The dataset might lack critical details such as trim levels, optional packages, and regional pricing adjustments, which can significantly influence the original price. For example, higher trims or optional luxury features could lead to inaccurate predictions if these factors are not explicitly represented.
- **Data Quality**: Insufficient representation of specific car models, particularly rare or high-end vehicles, in the dataset can lead to biased predictions. The lack of comprehensive data for older or less common models makes extrapolation more error-prone.
- **Model Assumptions**: The Random Forest model, while effective for many applications, may not capture the distinct dynamics of new car pricing. Extrapolation from used car prices, which are influenced by depreciation, may result in systematic errors.
- **Depreciation Curve**: Real-world depreciation trends vary widely across brands, models, and market conditions. Assumptions about these trends, even if based on general industry standards, may not apply universally, leading to deviations in predicted prices.

Supplementary Sections

Histogram





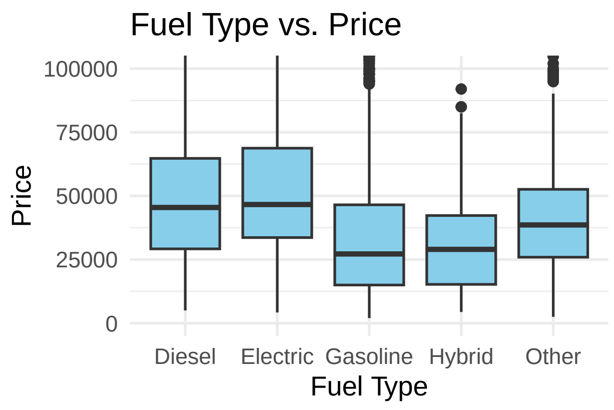


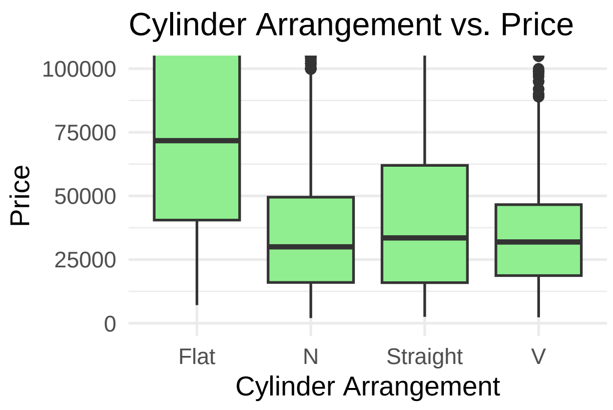
Frequency table

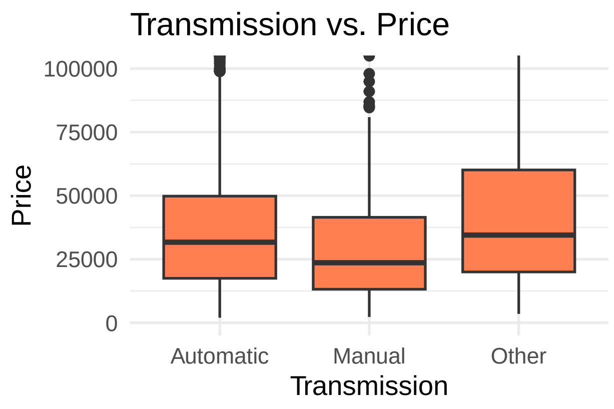
```
## $brand
##
##      Cheap     Luxury Mainstream      Other
##       1441       1760        709         99
##
## $fuel_type
##
##   Diesel Electric Gasoline   Hybrid    Other
##      114      202     2753       87      853
##
## $transmission
##
## Automatic    Manual     Other
##      3211       372       426
##
```

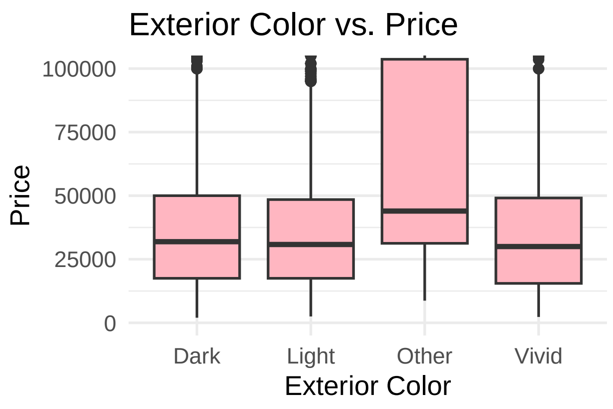
```
## $ext_col
##
##  Dark Light Other Vivid
##  1079  1952    94   884
##
## $int_col
##
##  Dark Light Other Vivid
##  2330  1178   291   210
##
## $model_year_dist
##
##      New      Old   Recent Very Old
##     1258      704     1367      680
##
## $milage_dist
##
##      High Mileage       Low Mileage  Moderate Mileage Very High Mileage
##              1117              1084              1512               296
##
## $price_dist
##
##         < $12,000         > $80,000 $12,000 - $21,999 $20,000 - $31,999
##               571               399               789               689
## $32,000 - $49,999 $50,000 - $79,999
##               976               585
```

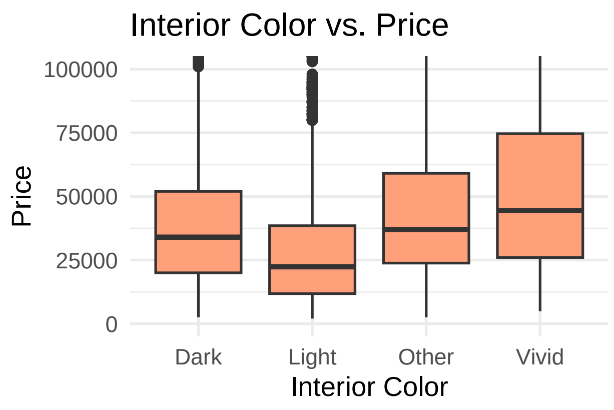
Boxplots

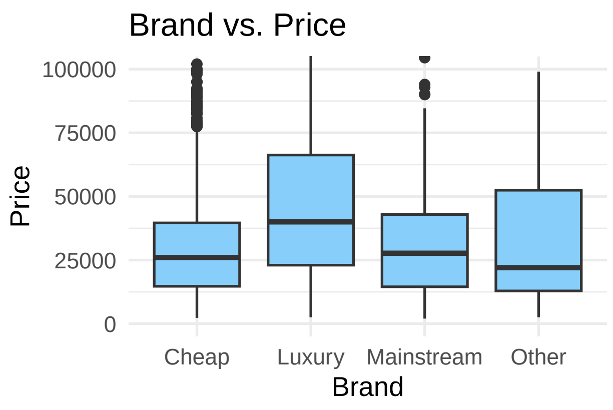

Fuel Type vs. Price
Price
100000
75000
50000
25000
0
Diesel
Electric
Gasoline
Hybrid
Other
Fuel Type


Cylinder Arrangement vs. Price
Price
100000
75000
50000
25000
0
Flat
N
Straight
V
Cylinder Arrangement

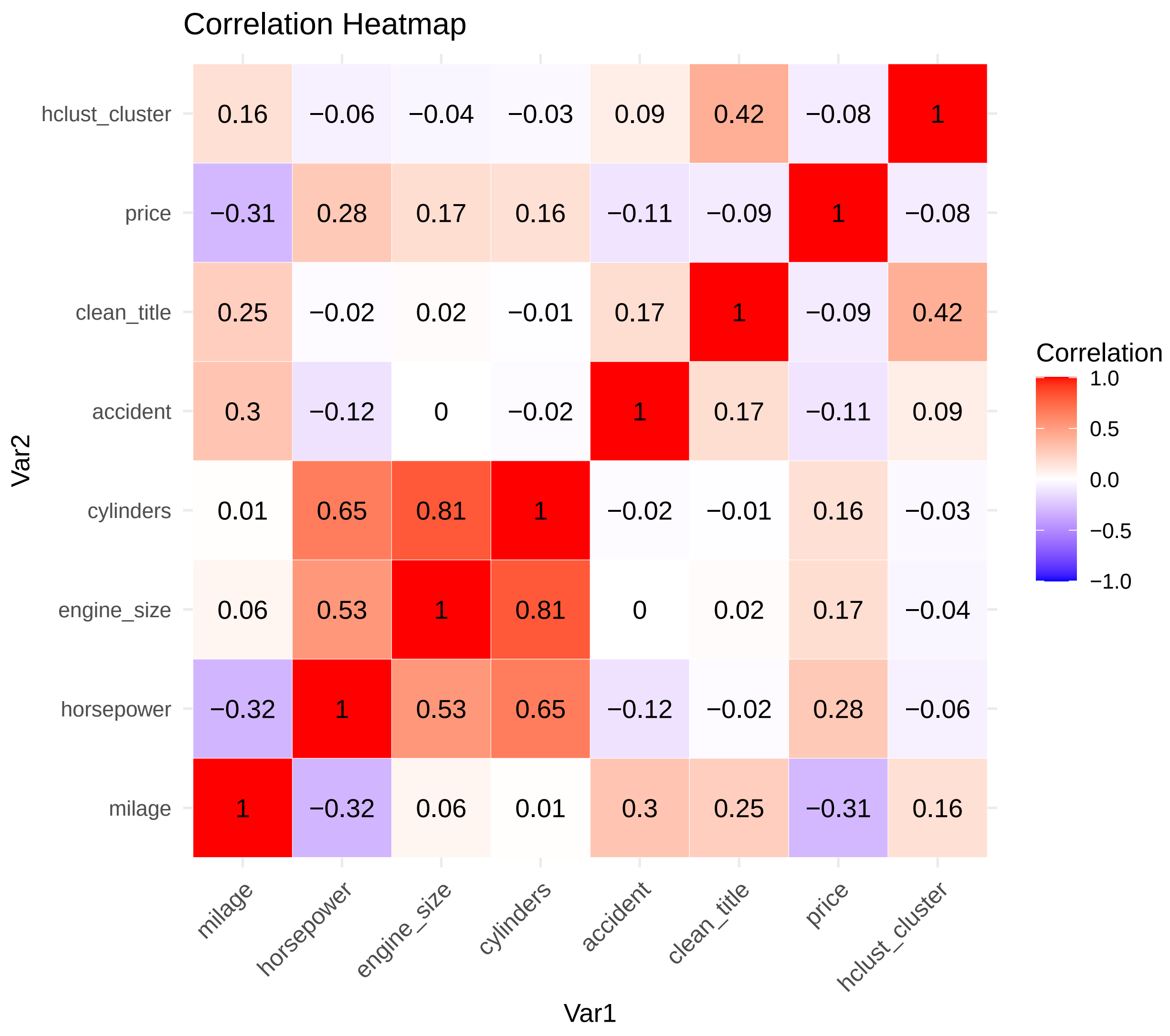

Transmission vs. Price
Price
100000
75000
50000
25000
0
Automatic
Manual
Other
Transmission

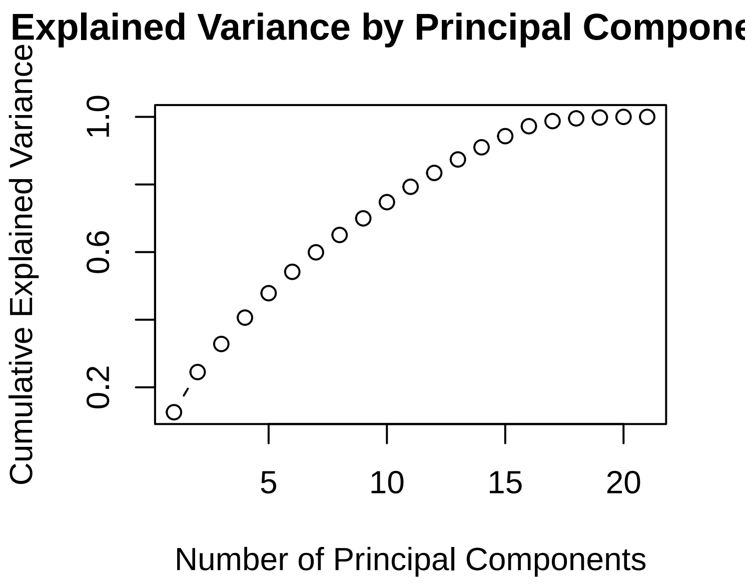

Exterior Color vs. Price
Price
100000
75000
50000
25000
0
Dark
Light
Other
Vivid
Exterior Color

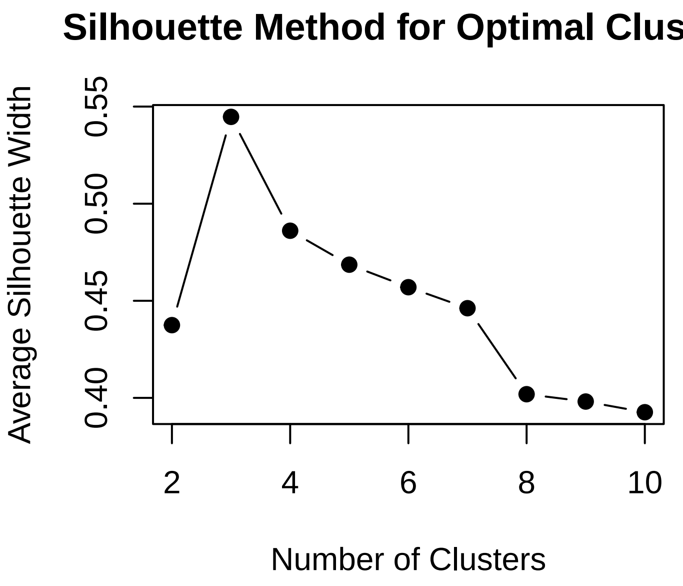

Interior Color vs. Price
Price
100000
75000
50000
25000
0
Dark
Light
Other
Vivid
Interior Color

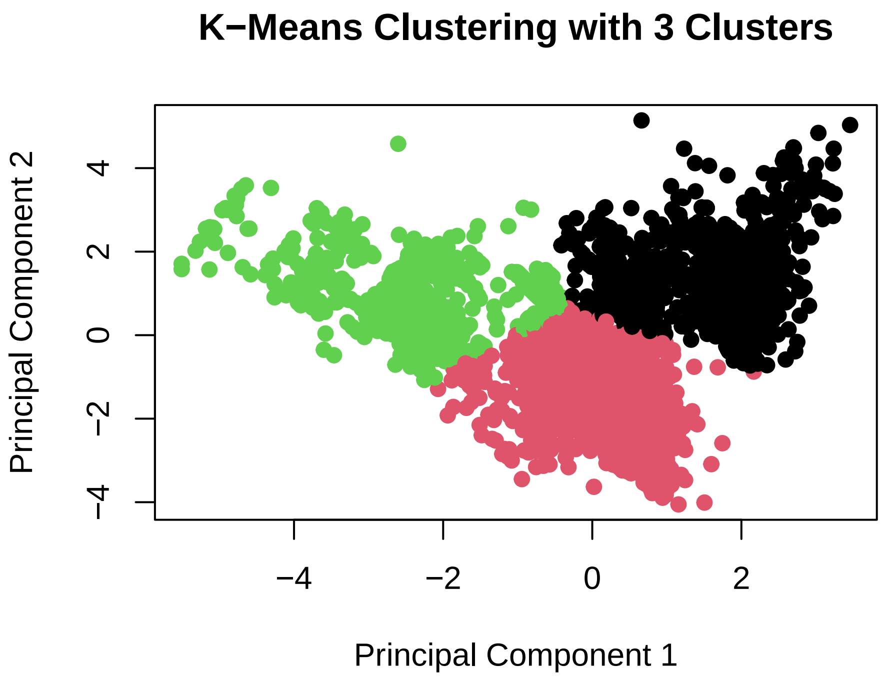

Brand vs. Price
Price
100000
75000
50000
25000
0
Cheap
Luxury
Mainstream
Other
Brand

Correlation Heatmap



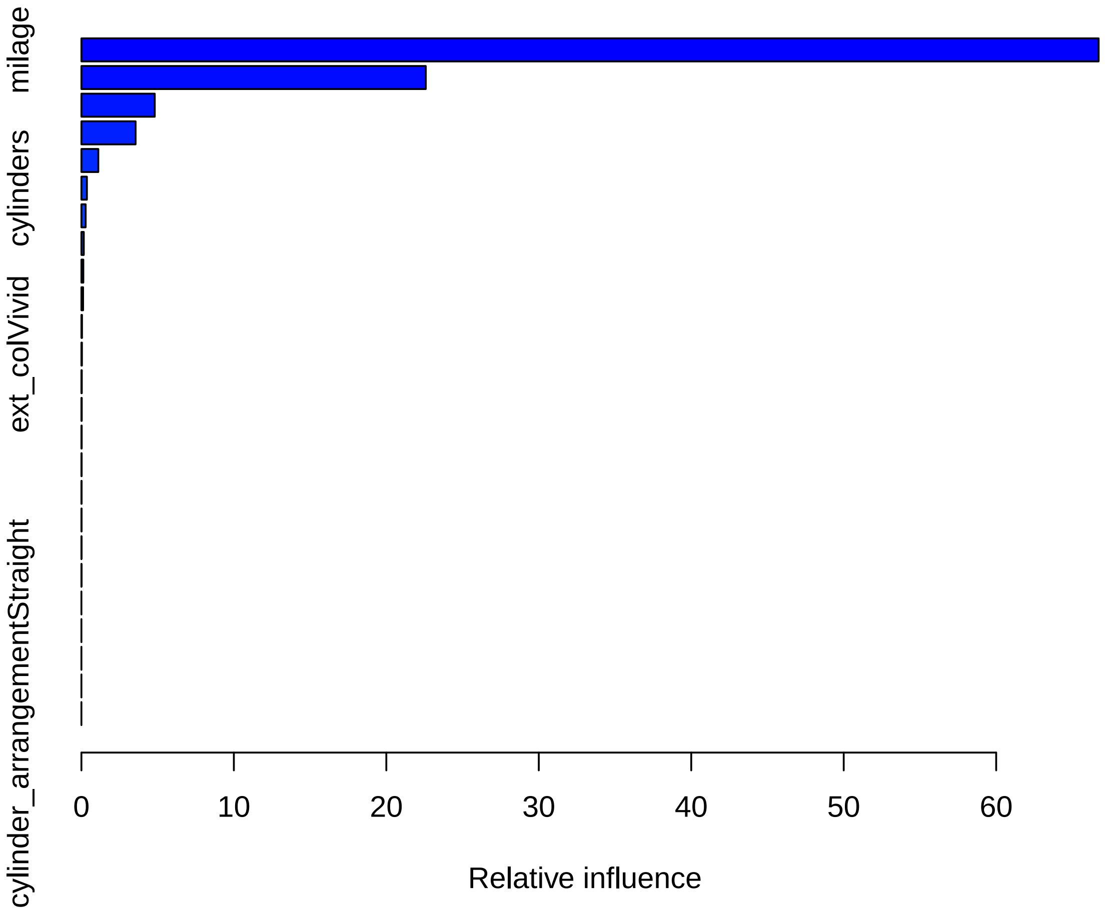






```
## [1] "Mean Silhouette Score for K-Means Clustering: 0.544706169415374"
```

Prediction Models(Model 3: GBM)



```
##                                                 var      rel.inf
## milage                                       milage 66.721079304
## horsepower                               horsepower 22.586462555
## engine_size                             engine_size  4.809474572
## PC2                                             PC2  3.553859826
## PC1                                             PC1  1.108903447
## cylinders                                 cylinders  0.360347875
## int_colOther                           int_colOther  0.277809945
## int_colLight                           int_colLight  0.155724715
## cylinder_arrangementN         cylinder_arrangementN  0.125277391
## fuel_typeGasoline                 fuel_typeGasoline  0.110196078
## cluster                                     cluster  0.039139873
## ext_colVivid                           ext_colVivid  0.034429256
## cylinder_arrangementV         cylinder_arrangementV  0.026706591
## fuel_typeOther                       fuel_typeOther  0.022541273
## ext_colOther                           ext_colOther  0.018997373
## transmissionOther                          transmissionOther  0.015652182
## accident                                            accident  0.014392673
## pca_cluster                                      pca_cluster  0.013025700
## fuel_typeElectric                          fuel_typeElectric  0.003124485
## ext_colLight                                    ext_colLight  0.002854887
## ext_colDark                                      ext_colDark  0.000000000
## int_colVivid                                    int_colVivid  0.000000000
## transmissionManual                        transmissionManual  0.000000000
## fuel_typeHybrid                              fuel_typeHybrid  0.000000000
## cylinder_arrangementStraight cylinder_arrangementStraight  0.000000000
```