Final Report C2

Analysis of BMW Auction Prices: Impacts of Mileage, Car Type, and Key Features

Team members (Lab C2): Ananya Agarwal, Sumanth Hosdurg Kamath, Sangwoo (Mark) Kim, Thi Phuong Thao Vu, Zhanbo Yang, Runqi Zhao

Abstract

This report explores the factors that affect prices at BMW car auctions using statistical modeling. Utilizing a data set containing 4,843 entries of BMWs over five years, we investigated how mileage, car type, and features impact prices. Linear regression, data transformation, and model refinement techniques were employed for this analysis to understand the relationship between these variables and auction prices. Initial results showed significant price variations based on car type, with coupes and SUVs commanding higher prices than convertibles. Mileage negatively impacted prices, with transformations enhancing model reliability. Additionally, including engine power in models increased their explanatory power from 42.42% to 53.1%. This study provides valuable insights for stakeholders in the automotive market, aiding price-setting strategies and informed purchasing decisions.

Introduction

Understanding the factors influencing car prices in the secondary market is essential for automotive stakeholders: including auction platforms, dealers, or consumers. This report will examine how mileage, car type, and key features impact BMW auction prices. Our main goal is to answer how much these factors collectively contribute to price variability and provide predictive models for price-setting and strategic buying. Specifically, we analyze: 1) The effect of car type on auction price. 2) The impact of mileage on price, considering car age. 3) The influence of specific features on price predictions.

By using linear regression models and enhancing them with data transformation and relevant variables, we will aim to create robust models with which price dynamics can be interpreted, strategic decisions guided, and the understanding of the automotive resale market increased.

Data

The dataset for this project includes essential variables characterizing BMW cars, such as mileage, engine_power, fuel type, paint_color, and car_type. It also contains boolean features (feature_1 to feature_8) that denote specific attributes, alongside sales information like car price and model_key. To provide additional insights, we created Car_Age, a new variable measuring the vehicle's age in years, calculated from the registration date to the sale date.

We improved data quality by filtering out implausible values, specifically records with negative or excessively high mileage (over 1,000,000 miles) and engine power less than or equal to zero. We further streamlined the data by removing redundant columns like registration_date and sold_at.

Furthermore, we categorized infrequent car models under "other models" and grouped less common car types into "other car types" based on frequency thresholds, simplifying the analysis while retaining meaningful information. (Table A1)

Explain the model for each variable

In Figure A1, the scatter plot shows square root-transformed car prices grouped by different car types, illustrating variations in price distribution across categories. SUVs and sedans generally have a wider spread and higher prices compared to other car types.

Figure A2 displays a scatter plot of mileage versus price, revealing a right-skewed distribution that necessitated square root transformations to normalize these variables. As a result, we applied square root transformations to both price and mileage to address this skewness.

In Figure A3, the scatter plot demonstrates a negative correlation between car age and the square root-transformed price, indicating that car prices decrease as vehicles age. This trend reflects depreciation over time, with older cars typically having lower prices, though significant variation exists among cars of the same age, suggesting other factors also play a role.

Figure A4 highlights feature patterns within the dataset, showing considerable variation in the presence of specific features among BMW cars. Common features like feature_1 and feature_7 may be standard or highly desirable, while less common features are likely exclusive to high-end models or optional packages, reflecting differences in car configurations and consumer preferences.

Lastly, Figure A5 reveals a positive relationship between engine power and the square root-transformed price, with higher engine power generally associated with higher car prices. However, the spread of data indicates that other factors also influence vehicle pricing.

Modeling and Analysis:

Stepwise Model

The variable paint color was dropped during stepwise regression due to its limited statistical significance and negligible contribution to model performance. While included, paint_color yielded high p-values for most levels, indicating weak relationships with the dependent variable, price_sqrt. Additionally, its inclusion resulted in an AIC value of 13252, which improved to 13242 upon removal. This signifies a more parsimonious model with better overall fit. Given that the primary focus was on predictors with substantial explanatory power, such as mileage and car type, retaining paint_color would have unnecessarily increased model complexity.

Model Notation:

$$\sqrt{price} = \beta_0 + \beta_1\sqrt{mileage} + \beta_2\ CarAge + \beta_3\ Car\ type_{Sedan} + \beta_4\ Car\ type_{SUV} + \beta_5\ Car\ type_{OtherCartypes} +$$

$$\beta_6\ Feature1 + \beta_7\ Feature2 + \beta_8\ Feature3 + \beta_9\ Feature4 + \beta_{10}\ Feature5 + \beta_{11}\ Feature6 + \beta_{12}\ Feature7 +$$

$$\beta_{13}\ Feature8+\beta_{14}\ engine\ power+\beta_{15}\ Fuel_{electro}+\beta_{16}\ Fuel_{hybrid\ petrol}+\beta_{17}\ Fuel_{petrol}+\beta_{18}\ Model\ Key_{118}+$$

$$\beta_{19}\ Model\ Key_{316}+\beta_{20}\ Model\ Key_{318}+\beta_{21}\ Model\ Key_{320}+\beta_{22}\ Model\ Key_{520}+\beta_{23}\ Model\ Key_{525}+$$

$$\beta_{24}\ Model\ Key_{530}+\beta_{25}\ Model\ Key_{X1}+\beta_{26}\ Model\ Key_{X3}+\beta_{27}\ Model\ Key_{X5}+\beta_{28}\ Model\ Key_{OtherModels}+\epsilon$$

Modeling and Analysis:

Interpretation for Stepwise Regression model Based Figure B1, on the residuals vs. fitted values plots, the residuals do not exhibit a clear pattern, with most points evenly scattered around the zero-mean line along the fitted values. In Figure B3, the square rooted standardized residual versus fitted value plot shows a slightly noticeable trend of residuals spanning out. In both of the plots, only a few points show large residuals compared to the majority. This suggests that the residual variance is generally constant, indicating that our model satisfies the assumption of homoscedasticity.

According to Figure B2, the normal Q-Q plot, our standardized residuals deviated from the 45 degree line at both ends of the quantiles. Specifically, the points deviate from the target line left of -2 theoretical quantiles and right of 2.5 theoretical quantiles. This means that our model does not satisfy the assumption of residuals being normally distributed very well.

Given Figure B4 the residuals versus the leverage plot, we can see that Cook's distance line has marked out no data point. Only 3 points have notably high standard residuals with zero to tiny leverage; 2 points have notably high leverages but very small standard residual values. In conclusion, no points exerting concerns of being outliers.

The residual standard error of our final model is 15.11 on 2404 degrees of freedom.

The multiple R^2 of the final model is 0.8027. The adjusted R^2 of the model is very close value of 0.8004. This means 80.27% of the variation in our data set is explained by our final model based on the multiple R^2, and 80.04% of the variation in our data set is explained by our final model based on the Adjusted R^2. Our model is a complex model with multiple variables, so the adjusted R-squared is a better indicator of model performance as it penalize complexity. Even so, our model has a very high coefficient of determination at 80%, which is a considered to be very good in general.

The F-test score is 349.4 on 28 and 2404 Degree of Freedom, which translates to a p-value less than 2.2 * 10^-16. We reject the null hypothesis of the F-test, meaning that at least one predictor significantly contribute to the explaining of variation in squared root of car price.

No predictors have adjusted variance inflation factors exceeding the commonly used thresholds of 5 or 10 (Highest for car type is 1.78). So we believe that multicollinearity is not an significant issue in this model. All predictors are within acceptable ranges after adjusting for degrees of freedom.

Interpretation for Lasso regression model The LASSO regression model retained most key predictors, such as mileage, car age, car type, features, engine power, and fuel types, while shrinking the coefficients of less important variables. However, it did not substantially reduce the number of variables, as most predictors remained with non-zero coefficients even at the optimal λ. This suggests that the dataset's structure and the relevance of predictors limited LASSO's ability to eliminate variables effectively.

Compare Stepwise Regression and Lasso regression The LASSO regression model did not significantly reduce the number of variables compared to the stepwise regression model, as both retained most key predictors, including mileage, car age, car type, features, engine power, and fuel types. While LASSO can shrink less important coefficients to zero, in this dataset, it primarily affected coefficient magnitudes without eliminating many variables. The stepwise regression model, guided by statistical criteria like AIC, had already excluded non-significant variables (e.g., paint color), achieving a similar predictive performance with an adjusted R^2 of 80%. Given this, LASSO did not simplify the model further or provide a notable advantage in prediction, likely due to the dataset's structure and the importance of most predictors.

Q1: How does car type impact auction price

In this model we reduced the car types from 8 to 4 types (Estate, SUV, Sedan and Other car type) based on the popularity of the car type.

From the visualization in the Training dataset (Figure A1), we can find the lowest average of 111 of the square root of the price from the estate, and the highest averaged square root of price of 142 from the SUV.

From the multivariable model fitted from the training dataset, car type showed a significant effect on the square root of price. Compared to the estate, the SUV has an average 33.4 increase in the square root of price, while the sedan only has an average increase of 9.6, and 7.11 for other car types. This indicates that, in the prediction of the car price, the car type is important, and considering all other situations (mileage, car age, features, etc) at the same level, the square root of the final price would be highest for SUV, and lowest for estate, and SUV would have an average 33.4 higher of the square root of price than estate.

From the stepwise model, we can find the car type played an important role in predicting the square root of price. Compared with the full model, if we remove the variable of reduced car type, the AIC would increase. The both side stepwise model suggests the removal the variable of paint color, which could decrease the AIC compared to the full model. And in this reduced model, we still have the car type as an important variable to include, which if we removed from the model, would still increase the AIC.

The final model after the stepwise selection still give high importance to the car type, but the effect size has changed. Compared to the estate, the SUV has an average 33.5 increase in the square root of price, while the sedan only has an average increase of 9.6, and 7.2 for other car types. This indicates that, the car type is still important in the prediction of car price from the reduced model. Considering all other situations (mileage, car age, features, ect) at the same level, the square root of the final price would be highest for the SUV, and lowest for the estate, and the SUV would bid for a 33.5 higher of the square root of price than estate, sedan would bid for a 9.6 higher than estate, and other car type would bid for a 7.2 higher than estate. The SUV would be the most profitable car type and would gain the highest returns from SUV and the least from the estate.

Q2: How does mileage affect price, and does this vary with car age?

As we have previously discovered in report 3 with the simple linear regression model that contains only mileage as the predicting variable and price as the response variable, the residual behavior did not meet the model assumption. Both the scatter plot of price vs mileage, and the diagnostic plots of the model points to heteroscedasticity with fanning out residual variance, and residuals deviate from normality. We attempted to fix these issues with square root transformation on both mileage and price, and the model improved.

The final model is also based on the findings of our second research question in the last report. The coefficient of the transformed mileage variable is -0.093690. This means that square root mileage has a

negative relationship with the square root price of the car, interpreted as one unit increase in squared rooted mileage of the car is equivalent to a 0.093690 decrease in the predicted value of the squared rooted price of the car.

The p-value of the coefficient of the square root of mileage is smaller than 2*10^(-16), which is statistically significant at $p < 0.001$. This suggests that the square root mileage predictor is significant in our final model of prediction of the squared root spruce of the auctioned car.

Q3:Comprehensive Analysis of Features 1 to 8 Impact on Car Price

The analysis highlights that features like feature_1, feature_2, feature_6, feature_7, and feature_8 are prominent, with feature_2 (79%) and feature_8 (54%) being especially significant. Feature_1 (55%) indicates a popular attribute, while feature_3 and feature_4 (20%) appear more specialized, likely limited to premium models. Feature_8 emerges as a key factor linked to higher prices, supported by its strong positive coefficient in the regression analysis.

The EDA visualizations provide insights into feature-price interactions. Convertibles, often associated with luxury models, show a wider price spread, potentially tied to features like feature_8. Mileage has the expected negative relationship with price, but this trend is linearized using square root transformations. Features like feature_1 and feature_6 are associated with lower mileage, further boosting prices, while feature_8 appears to retain value even in older cars.

Regression analysis reinforces the importance of features 1 to 8. Feature_1 ($\beta = 4.39$) and feature_2 ($\beta = 2.08$) show positive impacts, while feature_8 ($\beta = 6.83$) is the most influential, likely representing a luxury characteristic. Feature_6 ($\beta = 3.82$) also significantly contributes to higher prices. High-priced cars are often associated with features like feature_8, feature_6, and feature_1.

Feature_8 likely represents a luxury or performance-enhancing attribute, significantly driving car prices. Features 1 and 2 also play vital roles, appealing to a broad range of buyers. While feature_5 has the smallest impact, it still adds value in combination with other favorable factors like low mileage or newer age.

The analysis confirms that features like feature_8, feature_1, and feature_6 have a substantial impact on car prices, making them crucial attributes in buyer

Prediction:

Validation Dataset Description:

The validation dataset, consisting of 2,407 BMW entries, closely mirrors the training dataset in terms of key variables, such as mileage, car age, and car type. The average mileage in the validation set is 70,580 miles, which is similar to the training set's 71,120 miles. The average car age is also comparable, at 5.1 years in the validation set versus 5.0 years in the training set. Additionally, the distribution of car types, with SUVs and sedans representing 22% and 24%, respectively, is consistent across both datasets. These similarities ensure that the validation dataset is appropriate for evaluating our model's predictive performance.

Visualizations:

Visualizations: We evaluated the model's performance on the validation dataset using scatter plots and histograms. The scatter plot comparing predicted and actual square root-transformed prices shows a strong

positive trend, with predicted values closely following the actual outcomes along the regression line (Figure D1). The histogram illustrates the residual distribution from the stepwise regression model, centered around zero, which suggests that the model's predictions are largely unbiased. The nearly symmetrical, bell-shaped distribution indicates that most prediction errors are minor, though a few outliers suggest occasional larger discrepancies (Figure D2).

Predictive Performance Statistics:

The stepwise regression model yielded strong performance metrics on the validation dataset. The Mean Absolute Error (MAE) was 9.14, signifying the average absolute difference between the predicted and actual square root-transformed prices. The Root Mean Squared Error (RMSE) was 15.08, reflecting the typical magnitude of prediction errors, and the R-squared value was 0.792, indicating that the model explained 79.2% of the variance in the square root-transformed prices. These results demonstrate the model's robust predictive capabilities while also pointing to opportunities for refinement, particularly in handling outliers and extreme values.

Discussion:

Our analysis of BMW auction prices highlights the significant impact of various predictors, including mileage, car type, engine power, and key features, on car valuation. We utilized both linear regression and stepwise model refinement to ensure robust predictions and insights. The model's performance, as evaluated on the validation dataset, demonstrated strong predictive power, with the stepwise regression model achieving an R-squared value of 0.792 and a Root Mean Squared Error (RMSE) of 15.08.

Key findings include the clear influence of car type, with SUVs consistently commanding higher prices compared to other types, and the critical role of mileage, which negatively correlates with price even after transformation. The square root transformations of both mileage and price improved model reliability by addressing skewed distributions. Features like feature_8, associated with luxury or performance, emerged as important determinants of car price, reflecting consumer preferences for high-end attributes.

While our models explained a substantial portion of price variation, some limitations persist, such as the potential effects of unobserved variables or external market conditions that our analysis did not capture. Future work could explore more sophisticated modeling approaches or incorporate additional data, such as market demand fluctuations or regional economic factors, to further enhance predictive accuracy.

Author Contribution Statement:

Ananya Agarwal: Conception and design, Visualization, Interpreting results, Writing (original draft), Writing (review and editing)
Sumanth Hosdurg Kamath: Formal analysis (coding), Visualization, Interpreting results, Writing (original draft), Writing (review and editing)
Sangwoo (Mark) Kim: Interpreting results, Writing (original draft)
Thi Phuong Thao Vu: Conception and design, Formal analysis (coding), Visualization, Writing (original draft), Writing (review and editing), Project Coordination
Zhanbo Yang: Interpreting results, Writing (original draft), Writing (review and editing)
Runqi Zhao: Conception and design, Formal analysis (coding), Visualization, Interpreting results, Writing (original draft), Writing (review and editing)

Appendix

Step 1 Load Data and Cleaning

```
bmw_data <- read.csv("BMWpricing_updated.csv") %>%
  filter(mileage >= 0 & mileage < 1000000 &
           # price <= 120000 &
           engine_power > 0) %>%
  mutate(Car_Age = time_length(mdy(sold_at) -
                                 mdy(registration_date),
                               "year")) %>%
  select(-c("registration_date", "sold_at")) %>%
  select(c("mileage", "engine_power", "fuel",
           "paint_color", "car_type", "Car_Age",
           "feature_1", "feature_2", "feature_3",
           "feature_4", "feature_5", "feature_6",
           "feature_7", "feature_8", "obs_type", "price", "model_key"))

bmw_data$price_sqrt <- sqrt(bmw_data$price)
bmw_data$mileage_sqrt <- sqrt(bmw_data$mileage)

# Calculate frequencies of model_key and car type
model_key_frequency <- table(bmw_data$model_key)
car_type_frequency <- table(bmw_data$car_type)

# Create a list of model_keys with frequency > 100, car type  1000
frequent_model_keys <- names(model_key_frequency[model_key_frequency > 100])
frequent_car_type <- names(car_type_frequency[car_type_frequency > 1000])

# Create new columns for model key and car type
bmw_data <- bmw_data %>% mutate(
  model_key_categorized = ifelse(bmw_data$model_key %in% frequent_model_keys,
                                 as.character(bmw_data$model_key),
                                 "other models"),
  car_type_reduced = ifelse(bmw_data$car_type %in% frequent_car_type,
                            bmw_data$car_type, "other car types")
)
```

Table A1

```
library(gtsummary)
bmw_data %>% select(-c("model_key", "car_type")) %>% tbl_summary(by = obs_type) %>% as_kable()
```

Characteristic	**Training** N = 2,433	**Validation** N = 2,407
mileage	140,991 (102,162, 175,485)	141,603 (103,331, 175,007)

Characteristic	**Training** N = 2,433	**Validation** N = 2,407
engine_power	120 (100, 137)	120 (100, 135)
fuel		
diesel	2,324 (96%)	2,314 (96%)
electro	2 (<0.1%)	1 (<0.1%)
hybrid_petrol	7 (0.3%)	1 (<0.1%)
petrol	100 (4.1%)	91 (3.8%)
paint_color		
beige	24 (1.0%)	17 (0.7%)
black	814 (33%)	817 (34%)
blue	358 (15%)	352 (15%)
brown	172 (7.1%)	169 (7.0%)
green	9 (0.4%)	9 (0.4%)
grey	588 (24%)	587 (24%)
orange	4 (0.2%)	2 (<0.1%)
red	28 (1.2%)	24 (1.0%)
silver	167 (6.9%)	162 (6.7%)
white	269 (11%)	268 (11%)
Car_Age	4.84 (4.00, 5.83)	4.83 (4.08, 5.83)
feature_1	1,362 (56%)	1,298 (54%)
feature_2	1,958 (80%)	1,880 (78%)
feature_3	509 (21%)	469 (19%)
feature_4	480 (20%)	481 (20%)
feature_5	1,101 (45%)	1,129 (47%)
feature_6	576 (24%)	593 (25%)
feature_7	2,280 (94%)	2,232 (93%)
feature_8	1,312 (54%)	1,307 (54%)
price	14,300 (10,900, 18,800)	14,000 (10,800, 18,600)
price_sqrt	120 (104, 137)	118 (104, 136)
mileage_sqrt	375 (320, 419)	376 (321, 418)
model_key_categorized		
116	177 (7.3%)	181 (7.5%)
118	77 (3.2%)	65 (2.7%)
316	109 (4.5%)	126 (5.2%)
318	281 (12%)	288 (12%)
320	376 (15%)	376 (16%)
520	320 (13%)	313 (13%)
525	91 (3.7%)	93 (3.9%)
530	86 (3.5%)	71 (2.9%)
other models	436 (18%)	431 (18%)
X1	140 (5.8%)	134 (5.6%)
X3	212 (8.7%)	226 (9.4%)
X5	128 (5.3%)	103 (4.3%)
car_type_reduced		
estate	802 (33%)	804 (33%)
other car types	516 (21%)	494 (21%)
sedan	583 (24%)	584 (24%)
suv	532 (22%)	525 (22%)

Step 2 Seperate Dataset

Table A1 comapre between Train data and Validation data

```
Training <- filter(bmw_data, obs_type == "Training")
Validation <- filter(bmw_data, obs_type == "Validation")
bmw_data %>% select(-c("model_key", "car_type")) %>%
  tbl_summary(by = obs_type,
              statistic = list(all_continuous() ~ "{mean} ({sd})")) %>%
  add_p() %>%
  as_kable()
```

```
## The following errors were returned during `as_kable()`:
## x For variable `paint_color` (`obs_type`) and "estimate", "p.value",
##   "conf.low", and "conf.high" statistics: FEXACT error 7(location). LDSTP=17760
##   is too small for this problem, (pastp=531.323, ipn_0:=ipoin[itp=538]=767,
##   stp[ipn_0]=531.05). Increase workspace or consider using
##   'simulate.p.value=TRUE'
```

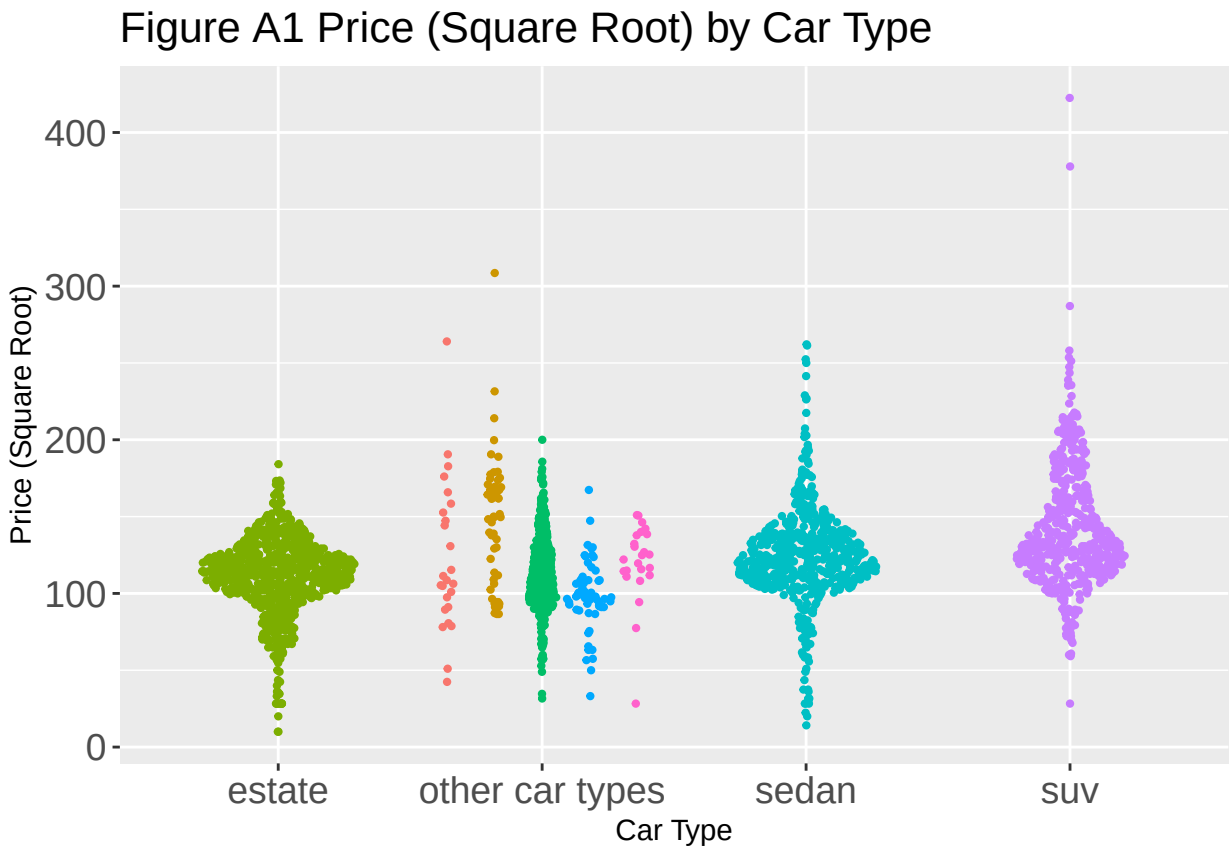
Characteristic	**Training** N = 2,433	**Validation** N = 2,407	**p-value**
mileage	140,592 (58,414)	141,063 (59,385)	>0.9
engine_power	130 (40)	128 (38)	0.3
fuel			0.14
diesel	2,324 (96%)	2,314 (96%)	
electro	2 (<0.1%)	1 (<0.1%)	
hybrid_petrol	7 (0.3%)	1 (<0.1%)	
petrol	100 (4.1%)	91 (3.8%)	
paint_color			
beige	24 (1.0%)	17 (0.7%)	
black	814 (33%)	817 (34%)	
blue	358 (15%)	352 (15%)	
brown	172 (7.1%)	169 (7.0%)	
green	9 (0.4%)	9 (0.4%)	
grey	588 (24%)	587 (24%)	
orange	4 (0.2%)	2 (<0.1%)	
red	28 (1.2%)	24 (1.0%)	
silver	167 (6.9%)	162 (6.7%)	
white	269 (11%)	268 (11%)	
Car_Age	5.42 (2.54)	5.44 (2.53)	0.6
feature_1	1,362 (56%)	1,298 (54%)	0.2
feature_2	1,958 (80%)	1,880 (78%)	0.042
feature_3	509 (21%)	469 (19%)	0.2
feature_4	480 (20%)	481 (20%)	0.8
feature_5	1,101 (45%)	1,129 (47%)	0.2
feature_6	576 (24%)	593 (25%)	0.4
feature_7	2,280 (94%)	2,232 (93%)	0.2
feature_8	1,312 (54%)	1,307 (54%)	0.8

Characteristic	Training N = 2,433	Validation N = 2,407	p-value
price	16,036 (9,844)	15,601 (8,455)	0.3
price_sqrt	122 (34)	120 (33)	0.3
mileage_sqrt	366 (82)	366 (83)	>0.9
model_key_categorized			0.8
116	177 (7.3%)	181 (7.5%)	
118	77 (3.2%)	65 (2.7%)	
316	109 (4.5%)	126 (5.2%)	
318	281 (12%)	288 (12%)	
320	376 (15%)	376 (16%)	
520	320 (13%)	313 (13%)	
525	91 (3.7%)	93 (3.9%)	
530	86 (3.5%)	71 (2.9%)	
other models	436 (18%)	431 (18%)	
X1	140 (5.8%)	134 (5.6%)	
X3	212 (8.7%)	226 (9.4%)	
X5	128 (5.3%)	103 (4.3%)	
car_type_reduced			>0.9
estate	802 (33%)	804 (33%)	
other car types	516 (21%)	494 (21%)	
sedan	583 (24%)	584 (24%)	
suv	532 (22%)	525 (22%)	

Training Dataset

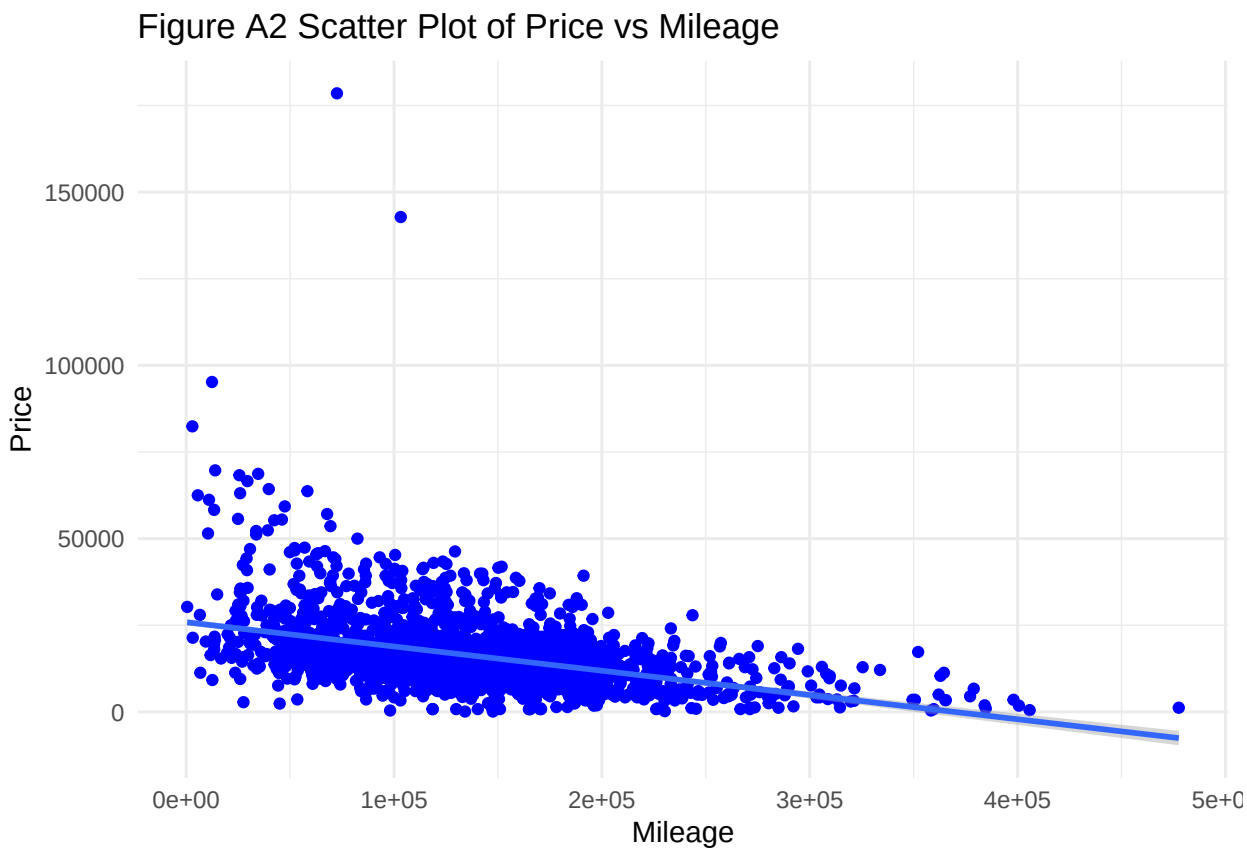
EDA - Visualization Figure A1-5

```
# Table
# Training %>% tbl_summary()

# Visualization
library(ggplot2)

# Car Type
Figure1 <- ggplot(Training, aes(x = car_type_reduced, y = price_sqrt, color = car_type)) +
  ggforce::geom_sina(size = 0.8) +
  labs(title = "Figure A1 Price (Square Root) by Car Type", x = "Car Type", y = "Price (Square
  theme(legend.position = "none",
        plot.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        legend.text = element_text(size = 12),
        legend.title = element_text(size = 14))
Figure1
```




```
# Mileage
Figure2A <- ggplot(Training, aes(x = mileage, y = price)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm") +
  labs(title = "Figure A2 Scatter Plot of Price vs Mileage",
       x = "Mileage",
       y = "Price") +
  theme_minimal()
Figure2A
```

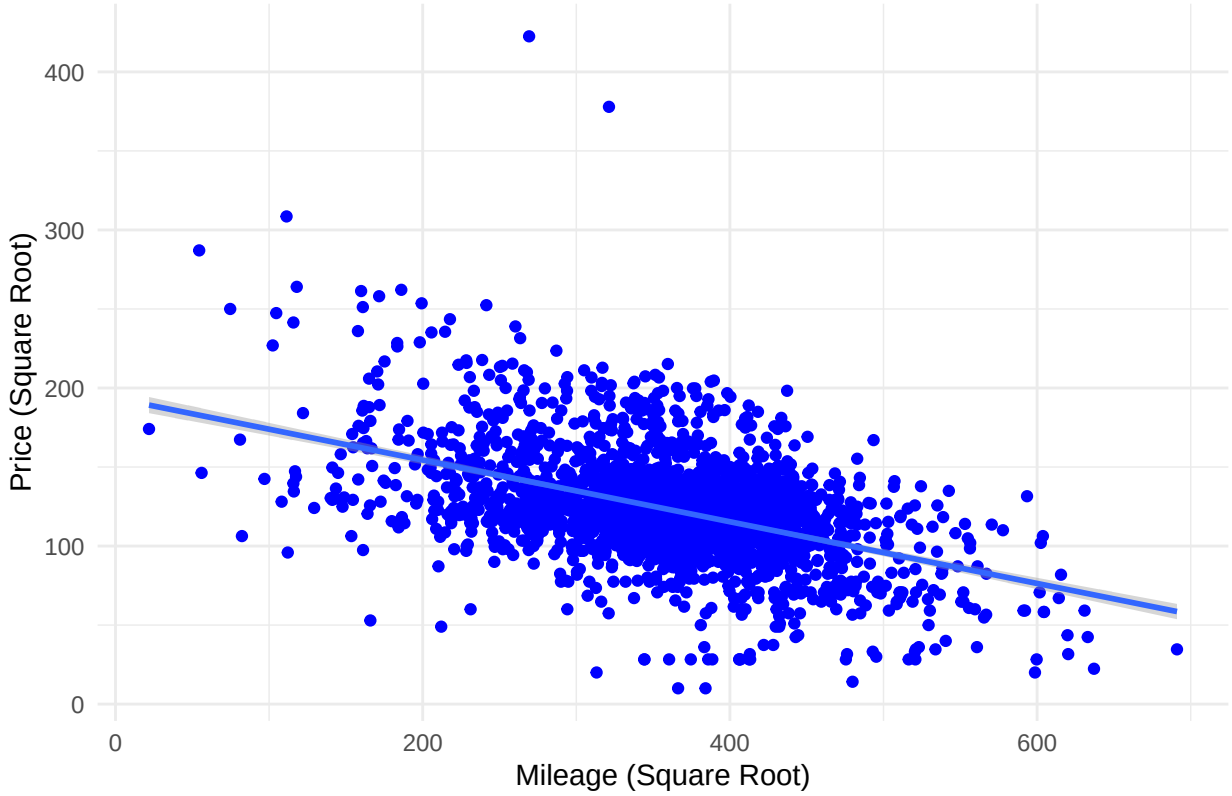
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
Figure2B <- ggplot(Training, aes(x = mileage_sqrt, y = price_sqrt)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm") +
  labs(title = "Figure A2 Scatter Plot of Price (Square Root) vs Mileage (Square Root)",
       x = "Mileage (Square Root)",
       y = "Price (Square Root)") +
  theme_minimal()
Figure2B
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

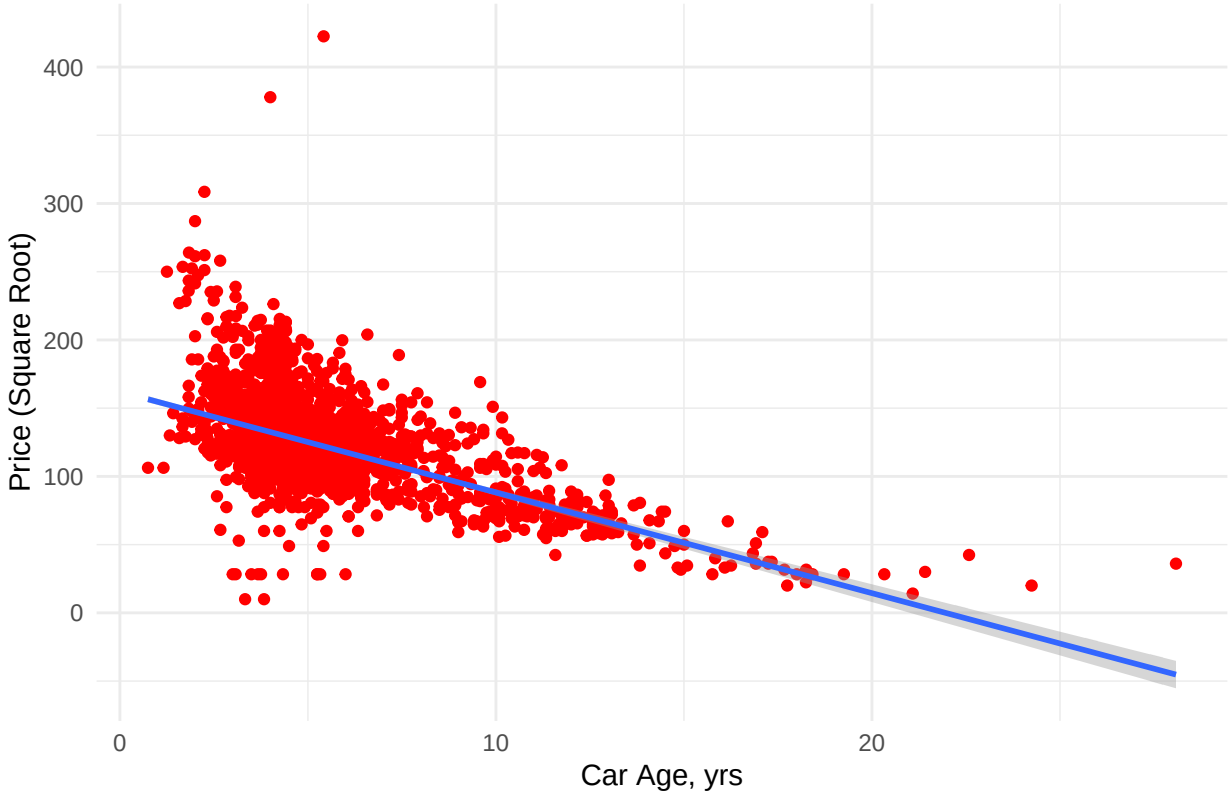
Figure A2 Scatter Plot of Price (Square Root) vs Mileage (Square Root)




```
# Car Age
Figure3 <- ggplot(Training, aes(x = Car_Age, y = price_sqrt)) +
  geom_point(color = "red") +
  geom_smooth(method = "lm") +
  labs(title = "Figure A3 Scatter Plot of Price (Square Root) vs Car_Age",
       x = "Car Age, yrs",
       y = "Price (Square Root)") +
  theme_minimal()
Figure3
```

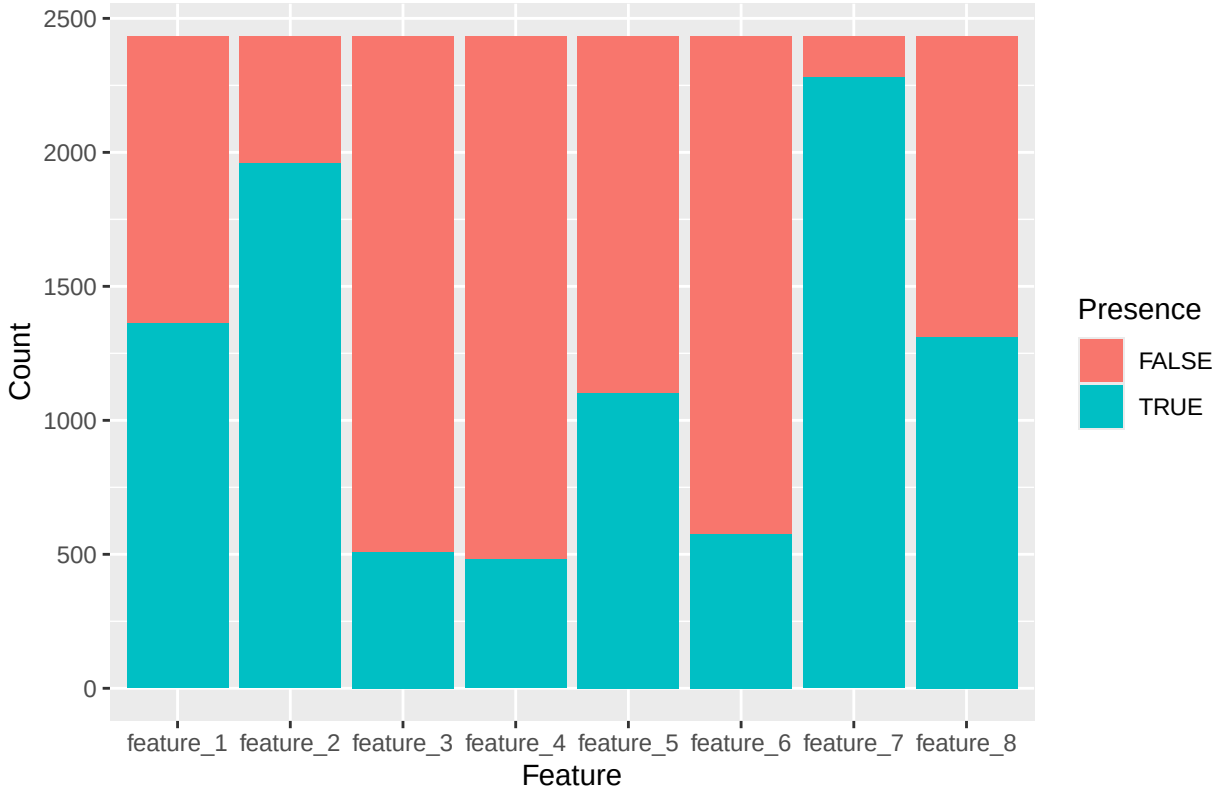
```
## `geom_smooth()` using formula = 'y ~ x'
```

Figure A3 Scatter Plot of Price (Square Root) vs Car_Age




```
# Features
Figure4 <- Training %>%
  select(starts_with("feature")) %>%
  gather(key = "Feature", value = "Presence") %>%
  ggplot(aes(x = Feature, fill = factor(Presence))) +
  geom_bar(position = "stack") +
  labs(title = "Figure A4 Distribution of Car Features", x = "Feature", y = "Count", fill = "P
Figure4
```

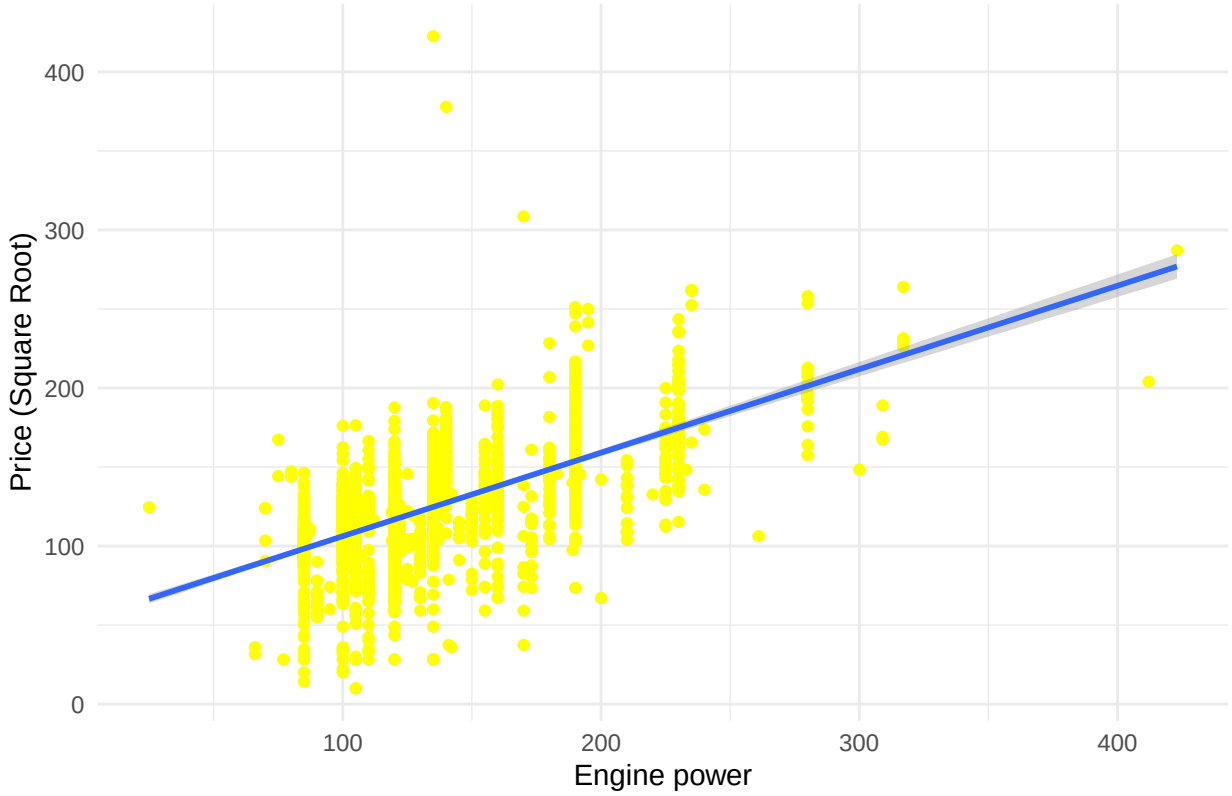
Figure A4 Distribution of Car Features




```
# Engine Power
Figure5 <- ggplot(Training, aes(x = engine_power, y = price_sqrt)) +
  geom_point(color = "yellow") +
  geom_smooth(method = "lm") +
  labs(title = "Figure A5 Scatter Plot of Price (Square Root) vs Engine power",
       x = "Engine power",
       y = "Price (Square Root)") +
  theme_minimal()
Figure5
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Figure A5 Scatter Plot of Price (Square Root) vs Engine power




```
# Combine
# grid.arrange(Figure1, Figure2A, Figure2B, Figure3, Figure4, Figure5, nrow = 3)
```

Full Model

```
Training_Full_Model <- lm(price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced +
                            feature_1 + feature_2 + feature_3 + feature_4 +
                            feature_5 + feature_6 + feature_7 + feature_8 +
                            engine_power + fuel + paint_color + model_key_categorized,
                          data = Training)
# summary(Training_Full_Model)
#
# # Set up plot layout
# par(mfrow = c(2, 2))
# plot(Training_Full_Model)
```

Stepwise Model

```
library(MLmetrics)
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
Training_Stepwise_model <- lm(price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced +
                              feature_1 + feature_2 + feature_3 + feature_4 +
                              feature_5 + feature_6 + feature_7 + feature_8 +
                              engine_power + fuel + paint_color + model_key_categorized,
                            data = Training)
stepwise_model_Train <- step(Training_Stepwise_model, direction = "both")
```

```
## Start:  AIC=13251.78
## price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced + feature_1 +
##     feature_2 + feature_3 + feature_4 + feature_5 + feature_6 +
##     feature_7 + feature_8 + engine_power + fuel + paint_color +
##     model_key_categorized
##
##                          Df Sum of Sq    RSS   AIC
## - paint_color             9      1907 548975 13242
## <none>                                 547068 13252
## - feature_2               1      1146 548213 13255
## - feature_4               1      1217 548285 13255
## - feature_5               1      1252 548320 13255
## - feature_7               1      1387 548455 13256
## - feature_3               1      1888 548956 13258
## - fuel                    3      5703 552771 13271
## - feature_6               1      5766 552834 13275
## - feature_1               1      8135 555203 13286
## - feature_8               1     17713 564781 13327
## - car_type_reduced        3     60289 607357 13500
## - model_key_categorized  11     72441 619509 13532
## - mileage_sqrt            1     93807 640875 13635
## - engine_power            1    104177 651245 13674
## - Car_Age                 1    222051 769119 14079
##
## Step:  AIC=13242.25
## price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced + feature_1 +
##     feature_2 + feature_3 + feature_4 + feature_5 + feature_6 +
##     feature_7 + feature_8 + engine_power + fuel + model_key_categorized
##
##                          Df Sum of Sq    RSS   AIC
## <none>                                 548975 13242
## - feature_4               1      1143 550118 13245
## - feature_2               1      1158 550133 13245
## - feature_5               1      1426 550401 13247
## - feature_7               1      1460 550436 13247
## - feature_3               1      1824 550799 13248
## + paint_color             9      1907 547068 13252
## - fuel                    3      5743 554718 13262
```

```
## - feature_6              1      5574 554549 13265
## - feature_1              1      8255 557230 13277
## - feature_8              1     17903 566878 13318
## - car_type_reduced       3     60857 609832 13492
## - model_key_categorized 11     73549 622524 13526
## - mileage_sqrt           1     93072 642047 13621
## - engine_power           1    105487 654462 13668
## - Car_Age                1    234727 783702 14106
```

```
stepwise_model_Train
```

```
##
## Call:
## lm(formula = price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced +
##     feature_1 + feature_2 + feature_3 + feature_4 + feature_5 +
##     feature_6 + feature_7 + feature_8 + engine_power + fuel +
##     model_key_categorized, data = Training)
##
## Coefficients:
##                         (Intercept)                         mileage_sqrt
##                           118.48271                             -0.09369
##                            Car_Age    car_type_reducedother car types
##                            -5.09710                              7.21005
##              car_type_reducedsedan                car_type_reducedsuv
##                             9.59330                             33.52800
##                      feature_1TRUE                      feature_2TRUE
##                             4.39354                             2.07617
##                      feature_3TRUE                      feature_4TRUE
##                             2.33224                             2.31684
##                      feature_5TRUE                      feature_6TRUE
##                             1.81765                             3.81699
##                      feature_7TRUE                      feature_8TRUE
##                             3.65473                             6.83055
##                       engine_power                        fuelelectro
##                             0.28003                            12.29853
##                  fuelhybrid_petrol                          fuelpetrol
##                            21.45268                            -5.22409
##          model_key_categorized118           model_key_categorized316
##                            -2.36673                             8.11177
##          model_key_categorized318           model_key_categorized320
##                             6.63363                             3.62955
##          model_key_categorized520           model_key_categorized525
##                            13.86773                             9.02847
##          model_key_categorized530  model_key_categorizedother models
##                             7.27373                            11.11168
##           model_key_categorizedX1            model_key_categorizedX3
##                           -21.49076                            -9.83796
##           model_key_categorizedX5
##                             4.31884
```

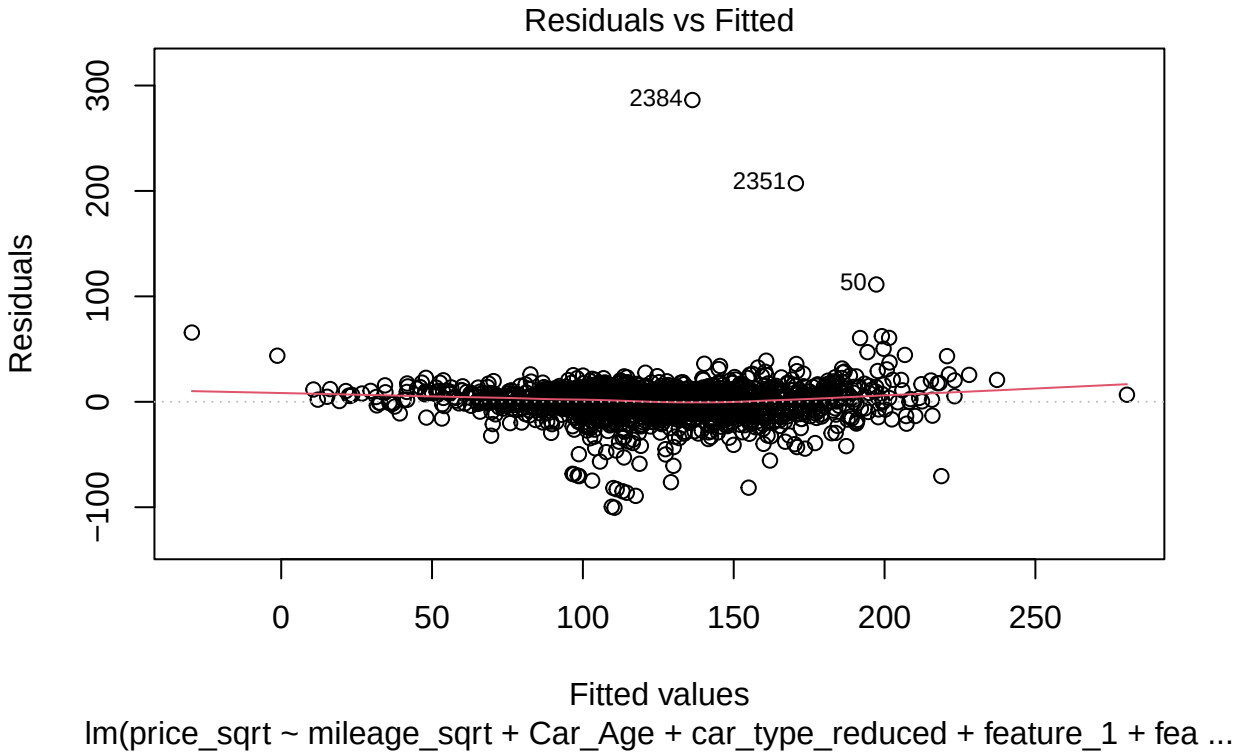
```r
Training_Stepwise_model <- lm(price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced +
                              feature_1 + feature_2 + feature_3 + feature_4 +
                              feature_5 + feature_6 + feature_7 + feature_8 +
                              engine_power + fuel + model_key_categorized,
                            data = Training)

summary(Training_Stepwise_model)
```

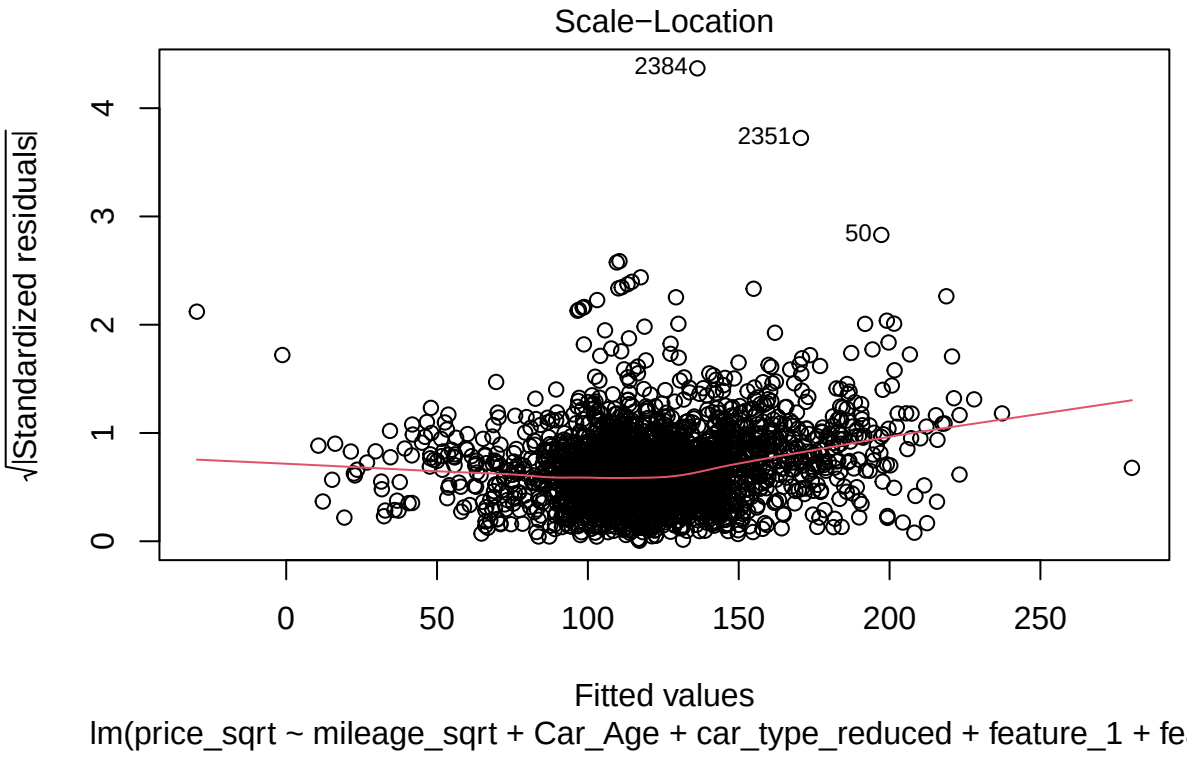
```
##
## Call:
## lm(formula = price_sqrt ~ mileage_sqrt + Car_Age + car_type_reduced +
##     feature_1 + feature_2 + feature_3 + feature_4 + feature_5 +
##     feature_6 + feature_7 + feature_8 + engine_power + fuel +
##     model_key_categorized, data = Training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -100.468   -5.269    0.892    6.735  286.199
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     118.482711   3.042689  38.940  < 2e-16 ***
## mileage_sqrt                     -0.093690   0.004641 -20.188  < 2e-16 ***
## Car_Age                          -5.097101   0.158983 -32.061  < 2e-16 ***
## car_type_reducedother car types   7.210047   1.462894   4.929 8.85e-07 ***
## car_type_reducedsedan             9.593305   0.854858  11.222  < 2e-16 ***
## car_type_reducedsuv              33.528004   2.425024  13.826  < 2e-16 ***
## feature_1TRUE                     4.393544   0.730757   6.012 2.11e-09 ***
## feature_2TRUE                     2.076166   0.921848   2.252 0.024401 *
## feature_3TRUE                     2.332241   0.825239   2.826 0.004750 **
## feature_4TRUE                     2.316837   1.035395   2.238 0.025336 *
## feature_5TRUE                     1.817650   0.727500   2.498 0.012539 *
## feature_6TRUE                     3.816989   0.772603   4.940 8.33e-07 ***
## feature_7TRUE                     3.654732   1.445158   2.529 0.011504 *
## feature_8TRUE                     6.830554   0.771436   8.854  < 2e-16 ***
## engine_power                      0.280034   0.013029  21.493  < 2e-16 ***
## fuelelectro                      12.298530  10.762529   1.143 0.253270
## fuelhybrid_petrol                21.452683   5.773489   3.716 0.000207 ***
## fuelpetrol                       -5.224090   1.705678  -3.063 0.002217 **
## model_key_categorized118         -2.366726   2.089329  -1.133 0.257424
## model_key_categorized316          8.111767   2.269352   3.574 0.000358 ***
## model_key_categorized318          6.633630   1.960597   3.383 0.000727 ***
## model_key_categorized320          3.629551   1.869696   1.941 0.052345 .
## model_key_categorized520         13.867733   1.979841   7.004 3.21e-12 ***
## model_key_categorized525          9.028473   2.461962   3.667 0.000251 ***
## model_key_categorized530          7.273733   2.575875   2.824 0.004785 **
## model_key_categorizedother models 11.111680   1.671345   6.648 3.66e-11 ***
## model_key_categorizedX1          -21.490755   2.839060  -7.570 5.30e-14 ***
## model_key_categorizedX3           -9.837962   2.731333  -3.602 0.000322 ***
```

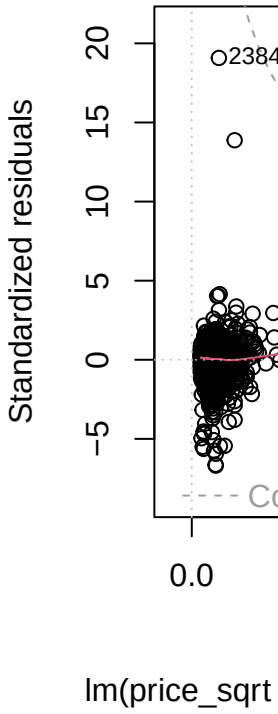
```
## model_key_categorizedX5             4.318840   2.952051   1.463 0.143599
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.11 on 2404 degrees of freedom
## Multiple R-squared:  0.8027, Adjusted R-squared:  0.8004
## F-statistic: 349.4 on 28 and 2404 DF,  p-value: < 2.2e-16
```

Figure B Diagnose of Stepwise model

```
# Set up plot layout
# par(mfrow = c(2, 2))
plot(Training_Stepwise_model)
```







Variance Inflation factors of Stepwise training Model

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(MLmetrics)
vif(Training_Stepwise_model)
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## mileage_sqrt         1.543134  1        1.242230
## Car_Age              1.738333  1        1.318458
## car_type_reduced    31.047437  3        1.772846
## feature_1            1.402013  1        1.184066
## feature_2            1.422541  1        1.192703
## feature_3            1.200390  1        1.095623
```

```
## feature_4              1.808820  1        1.344924
## feature_5              1.397003  1        1.181949
## feature_6              1.149176  1        1.071996
## feature_7              1.311285  1        1.145113
## feature_8              1.575358  1        1.255133
## engine_power           2.930369  1        1.711832
## fuel                   1.260130  3        1.039288
## model_key_categorized 79.299723 11        1.219918
```

Lasso Regression

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

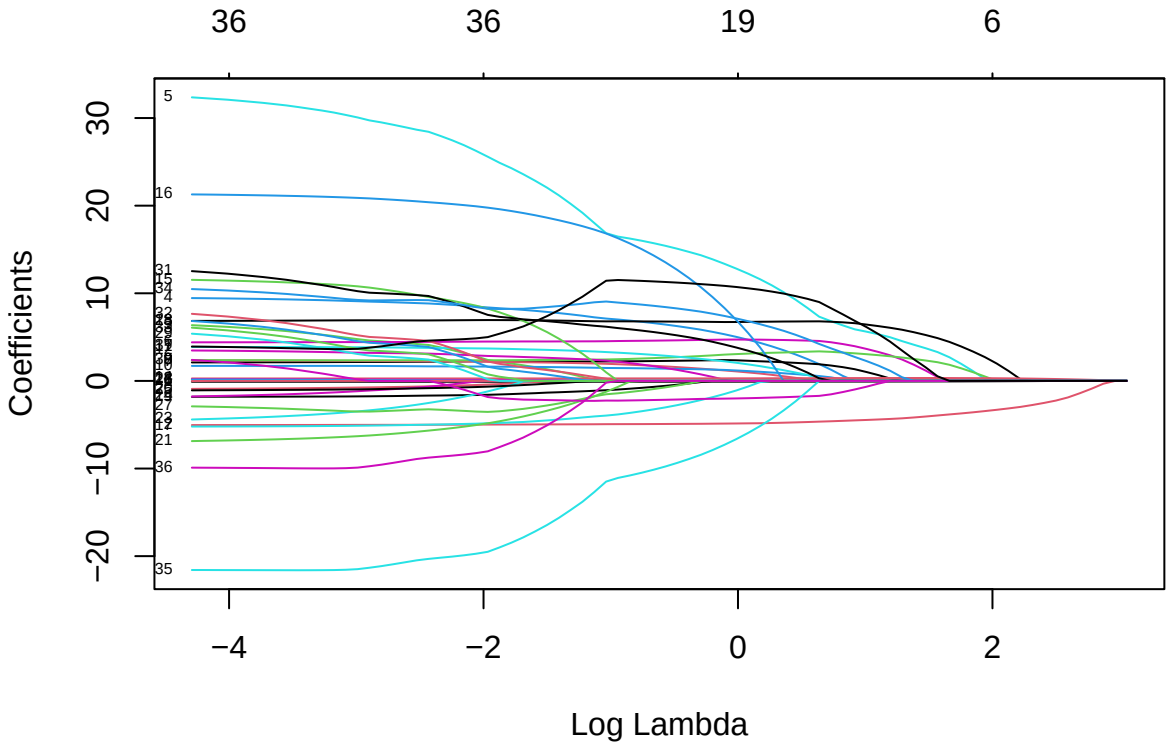
```
## Loaded glmnet 4.1-8
```

```
library(nlme)
```

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
lasso_model <- glmnet(x = model.matrix(Training_Full_Model)[,-1],
                      y = Training$price_sqrt,
                      alpha = 1)
plot(lasso_model, xvar = "lambda", label = TRUE)
```




```
write.csv(data.frame(Df = lasso_model$df, Dev_pct = lasso_model$dev.ratio*100, Lambda = lasso_
```

```
cv_model <- cv.glmnet(x = model.matrix(Training_Full_Model)[,-1],
                      y = Training$price_sqrt,
                      alpha = 1)
best_lambda <- cv_model$lambda.min
cat("Best Lambda - LASSO:", best_lambda)
```

```
## Best Lambda - LASSO: 0.01366951
```

```
lasso_coef <- coef(lasso_model, s = best_lambda)
# Summarize the LASSO model
print(lasso_coef)
```

```
## 38 x 1 sparse Matrix of class "dgCMatrix"
##                                           s1
## (Intercept)                      120.29499286
## mileage_sqrt                      -0.09435605
## Car_Age                           -5.04107194
## car_type_reducedother car types    6.36053443
## car_type_reducedsedan              9.45299924
## car_type_reducedsuv               32.36146152
## feature_1TRUE                      4.40435783
## feature_2TRUE                      2.10889478
```

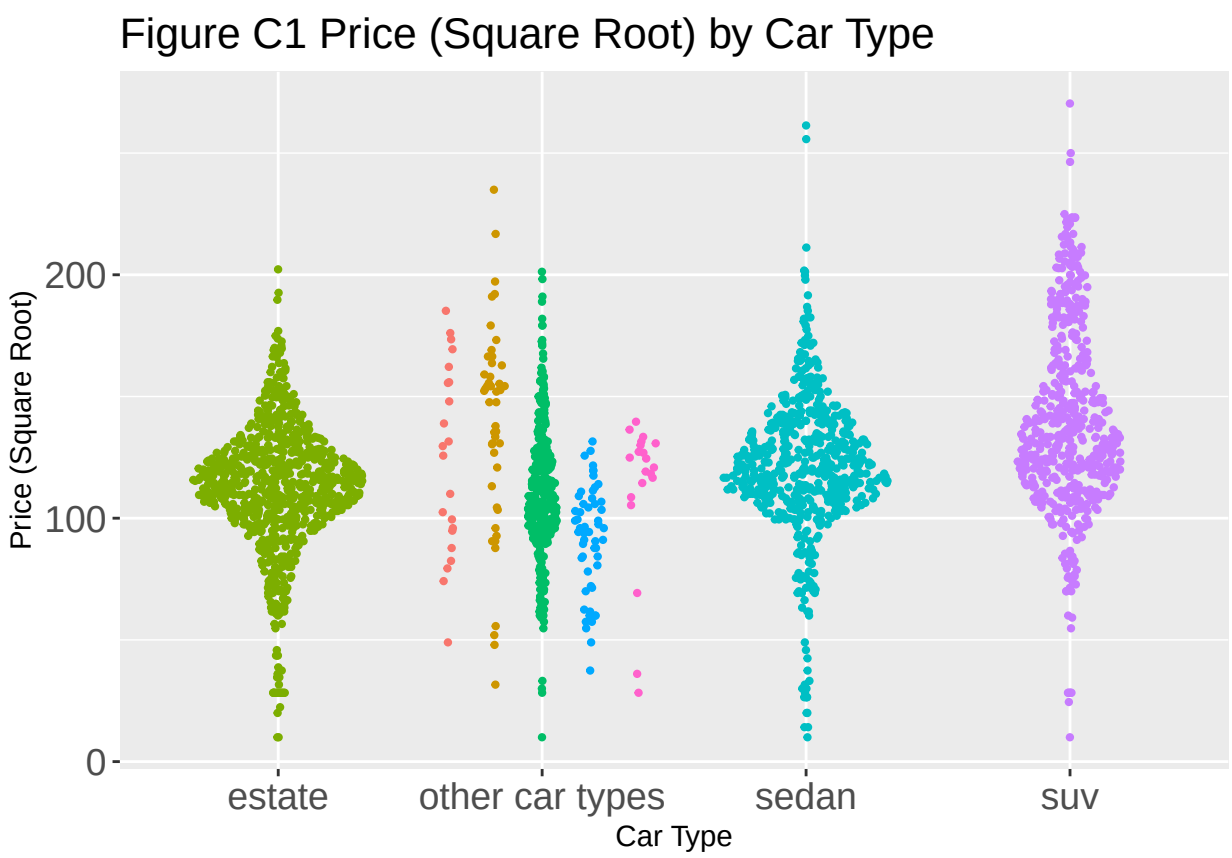
```
## feature_3TRUE                          2.34602416
## feature_4TRUE                          2.37088210
## feature_5TRUE                          1.71129649
## feature_6TRUE                          3.87818234
## feature_7TRUE                          3.47271672
## feature_8TRUE                          6.84655209
## engine_power                           0.27948272
## fuelelectro                           11.53430026
## fuelhybrid_petrol                     21.28524362
## fuelpetrol                            -5.20875083
## paint_colorblack                       0.19341347
## paint_colorblue                       -1.81466645
## paint_colorbrown                      -0.91117540
## paint_colorgreen                      -6.86907239
## paint_colorgrey                        0.14906873
## paint_colororange                     -4.40337393
## paint_colorred                        -1.77855066
## paint_colorsilver                     -1.07949887
## paint_colorwhite                      -0.01925398
## model_key_categorized118              -2.90278318
## model_key_categorized316               6.83867531
## model_key_categorized318               5.38969872
## model_key_categorized320               2.39847475
## model_key_categorized520              12.53022396
## model_key_categorized525               7.65662399
## model_key_categorized530               6.07594007
## model_key_categorizedother models     10.48164474
## model_key_categorizedX1              -21.57977223
## model_key_categorizedX3               -9.90089122
## model_key_categorizedX5                3.92292676
```

Validation

EDA - Visualization

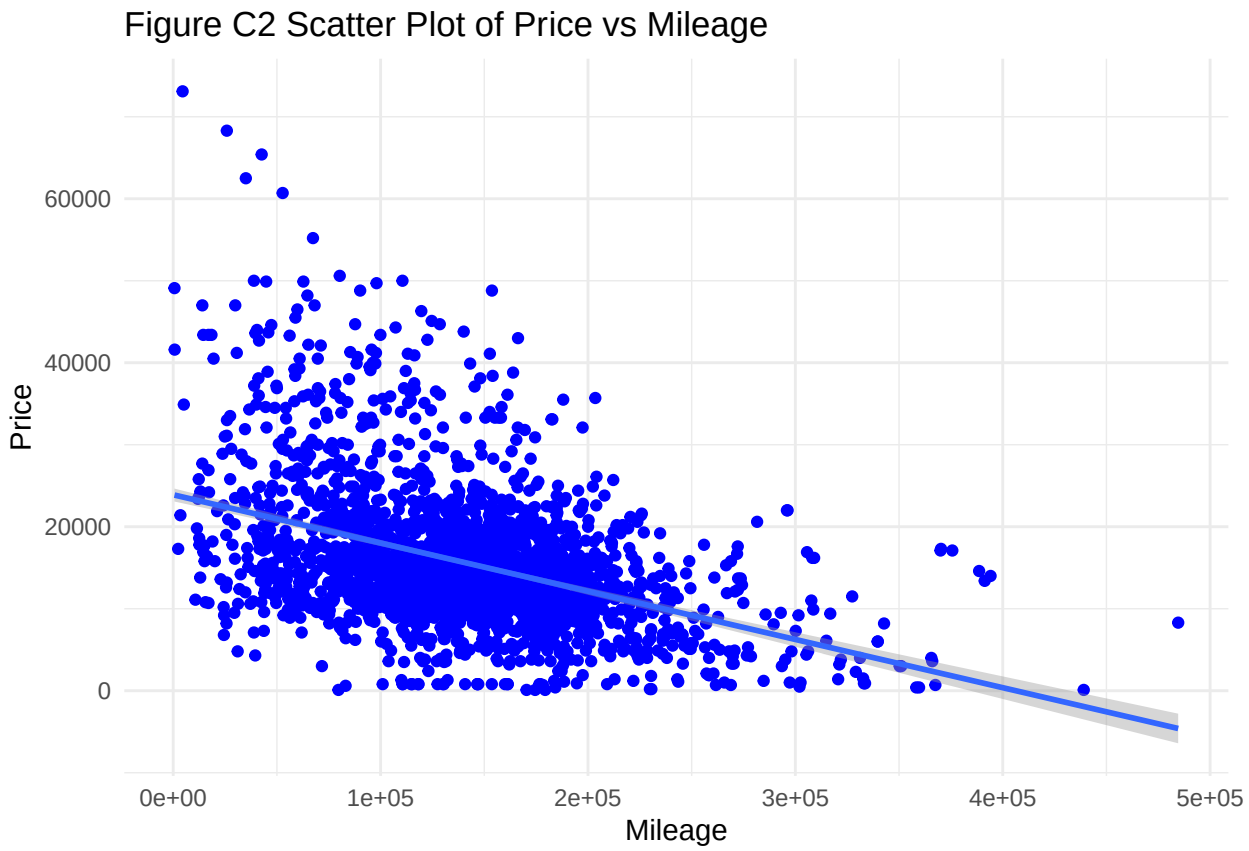
```
# Table
# Validation %>% tbl_summary()

# Visualization
# Car Type
Figure1 <- ggplot(Validation, aes(x = car_type_reduced, y = price_sqrt, color = car_type)) +
  ggforce::geom_sina(size = 0.8) +
  labs(title = "Figure C1 Price (Square Root) by Car Type", x = "Car Type", y = "Price (Square
  theme(legend.position = "none",
        plot.title = element_text(size = 16),
        axis.text = element_text(size = 14),
        legend.text = element_text(size = 12),
```

```
        legend.title = element_text(size = 14))
Figure1
```




```
# Mileage
Figure2A <- ggplot(Validation, aes(x = mileage, y = price)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm") +
  labs(title = "Figure C2 Scatter Plot of Price vs Mileage",
       x = "Mileage",
       y = "Price") +
  theme_minimal()
Figure2A
```

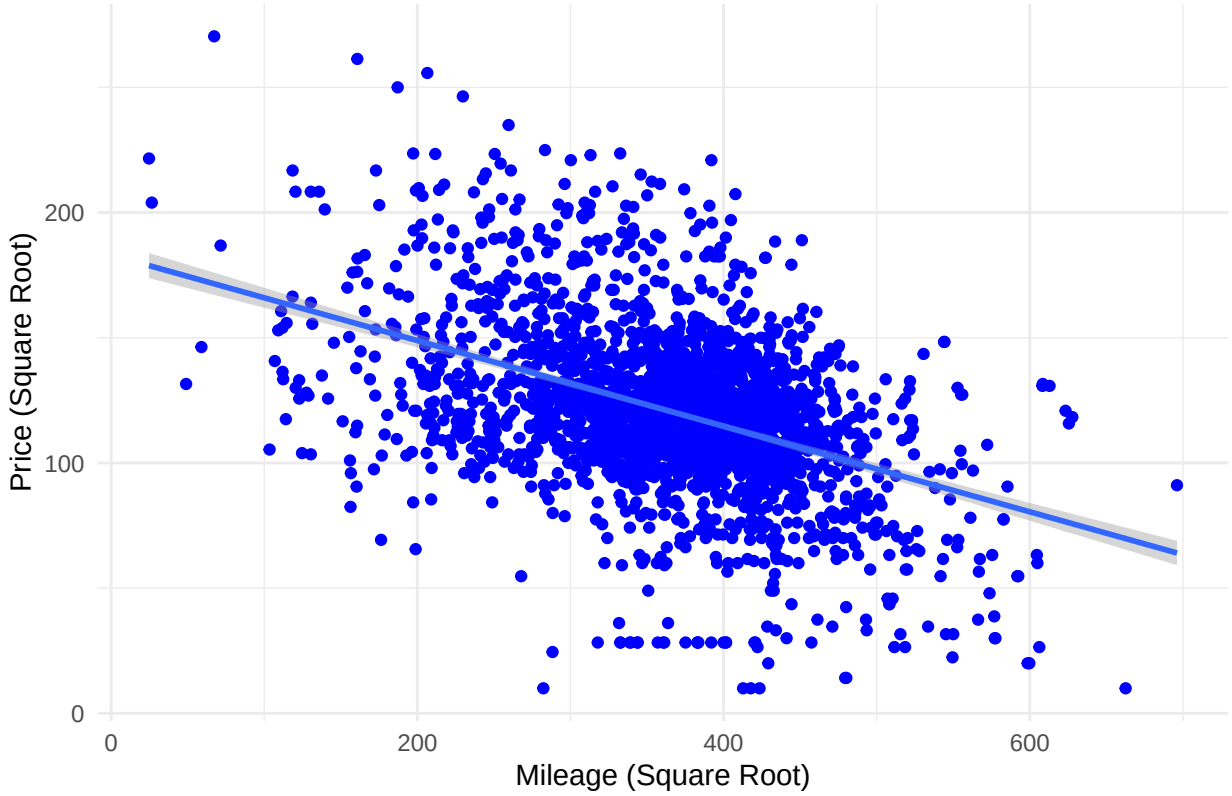
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
Figure2B <- ggplot(Validation, aes(x = mileage_sqrt, y = price_sqrt)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm") +
  labs(title = "Figure C2 Scatter Plot of Price (Square Root) vs Mileage (Square Root)",
       x = "Mileage (Square Root)",
       y = "Price (Square Root)") +
  theme_minimal()
Figure2B
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

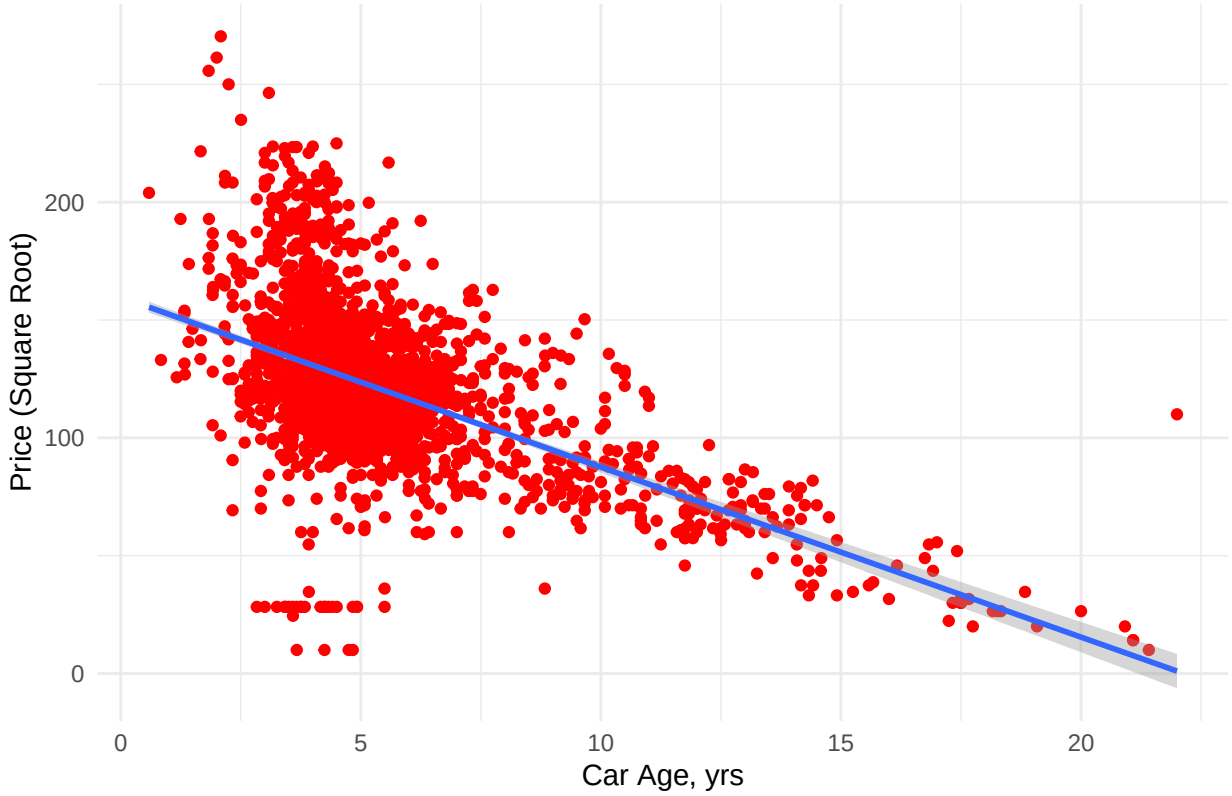
Figure C2 Scatter Plot of Price (Square Root) vs Mileage (Square Root)




```
# Car Age
Figure3 <- ggplot(Validation, aes(x = Car_Age, y = price_sqrt)) +
  geom_point(color = "red") +
  geom_smooth(method = "lm") +
  labs(title = "Figure C3 Scatter Plot of Price (Square Root) vs Car_Age",
       x = "Car Age, yrs",
       y = "Price (Square Root)") +
  theme_minimal()
Figure3
```

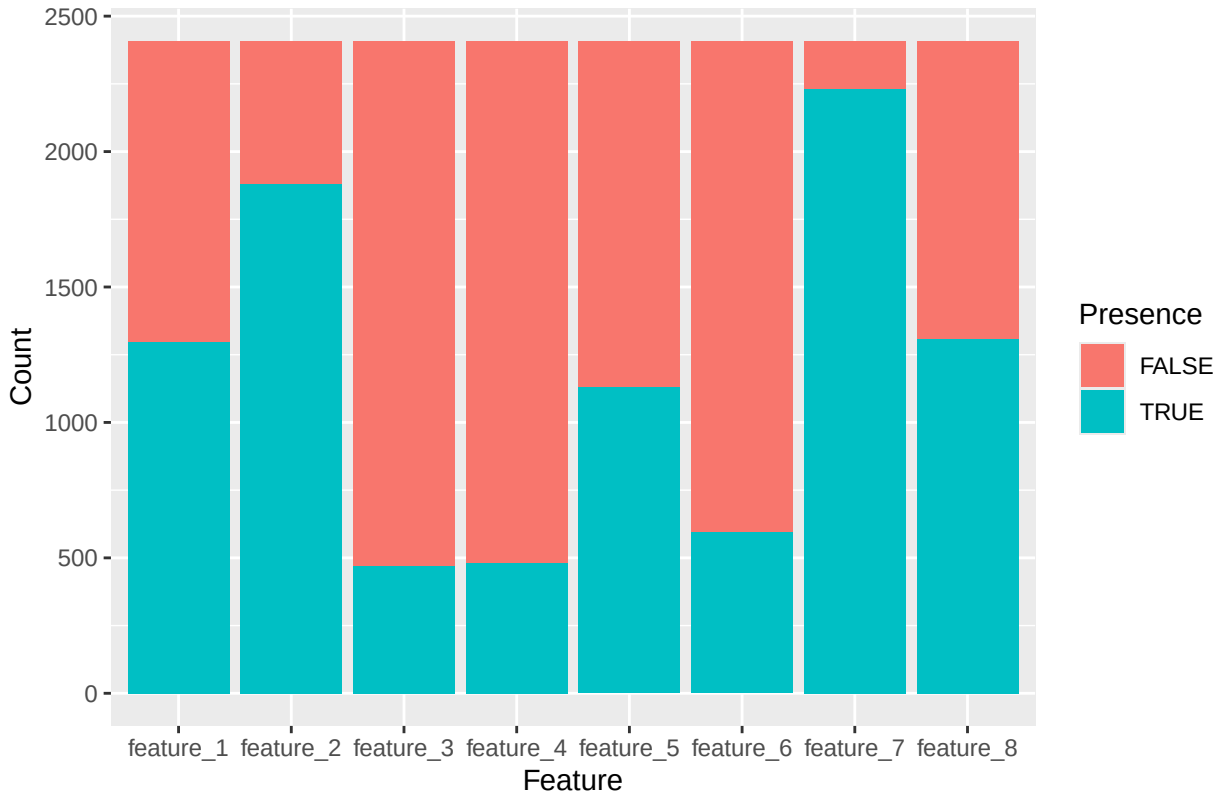
```
## `geom_smooth()` using formula = 'y ~ x'
```

Figure C3 Scatter Plot of Price (Square Root) vs Car_Age




```
# Features
Figure4 <- Validation %>%
  select(starts_with("feature")) %>%
  gather(key = "Feature", value = "Presence") %>%
  ggplot(aes(x = Feature, fill = factor(Presence))) +
  geom_bar(position = "stack") +
  labs(title = "Figure C4 Distribution of Car Features", x = "Feature", y = "Count", fill = "P
Figure4
```

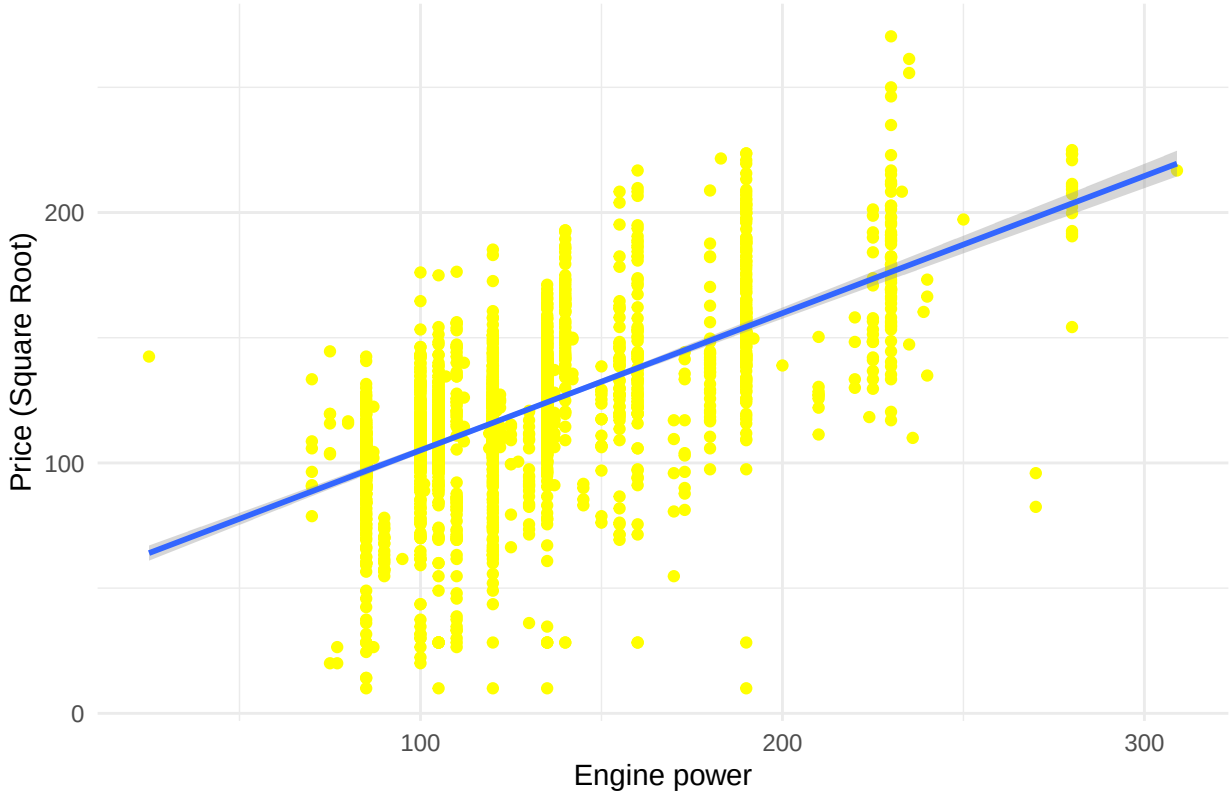
Figure C4 Distribution of Car Features




```
# Engine Power
Figure5 <- ggplot(Validation, aes(x = engine_power, y = price_sqrt)) +
  geom_point(color = "yellow") +
  geom_smooth(method = "lm") +
  labs(title = "Figure C5 Scatter Plot of Price (Square Root) vs Engine power",
       x = "Engine power",
       y = "Price (Square Root)") +
  theme_minimal()
Figure5
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Figure C5 Scatter Plot of Price (Square Root) vs Engine power




```
# Combine
# grid.arrange(Figure1, Figure2A, Figure2B, Figure3, Figure4, Figure5, nrow = 3)
```

Full Model - Prediction on Validation Dataset

```
# Validation$Predicted_Price_Sqrt_Full <- predict(Training_Full_Model, newdata = Validation)
# Validation$Pred_Full_Original <- Validation$Predicted_Price_Sqrt_Full - Validation$price_s
#
# ggplot(Validation, aes(x = price_sqrt, y = Predicted_Price_Sqrt_Full)) +
#   geom_point() +
#   geom_smooth(method = "lm") +
#   labs(title = "Scatter Plot of Price (Square Root) vs Predicted Price (Square Root)",
#        x = "Price (Square Root)",
#        y = "Predicted Price (Square Root)")
#
# ggplot(Validation, aes(x = Pred_Full_Original)) +
#   geom_histogram() +
#   labs(title = "Histogram of difference between Predicted Price (Square Root) to Price (Sq
#
# # Calculate performance metrics: MAE, RMSE, and R-squared
# mae <- mean(abs(Validation$Predicted_Price_Sqrt_Full - Validation$price_sqrt))
# rmse <- sqrt(mean((Validation$Predicted_Price_Sqrt_Full - Validation$price_sqrt)^2))
#
# # Calculate R-squared
```

```r
# rss <- sum((Validation$Predicted_Price_Sqrt_Full - Validation$price_sqrt)^2)  # Residual s
# tss <- sum((Validation$price_sqrt - mean(Validation$price_sqrt))^2)       # Total sum of s
# rsq <- 1 - (rss / tss)  # R-squared
#
# # Print performance metrics
# cat("Validation Set Performance Metrics for Full Model:\n")
# cat("Mean Absolute Error (MAE):", mae, "\n")
# cat("Root Mean Squared Error (RMSE):", rmse, "\n")
# cat("R-squared (R²):", rsq, "\n")
```

Stepwise Model - Prediction on Validation Dataset

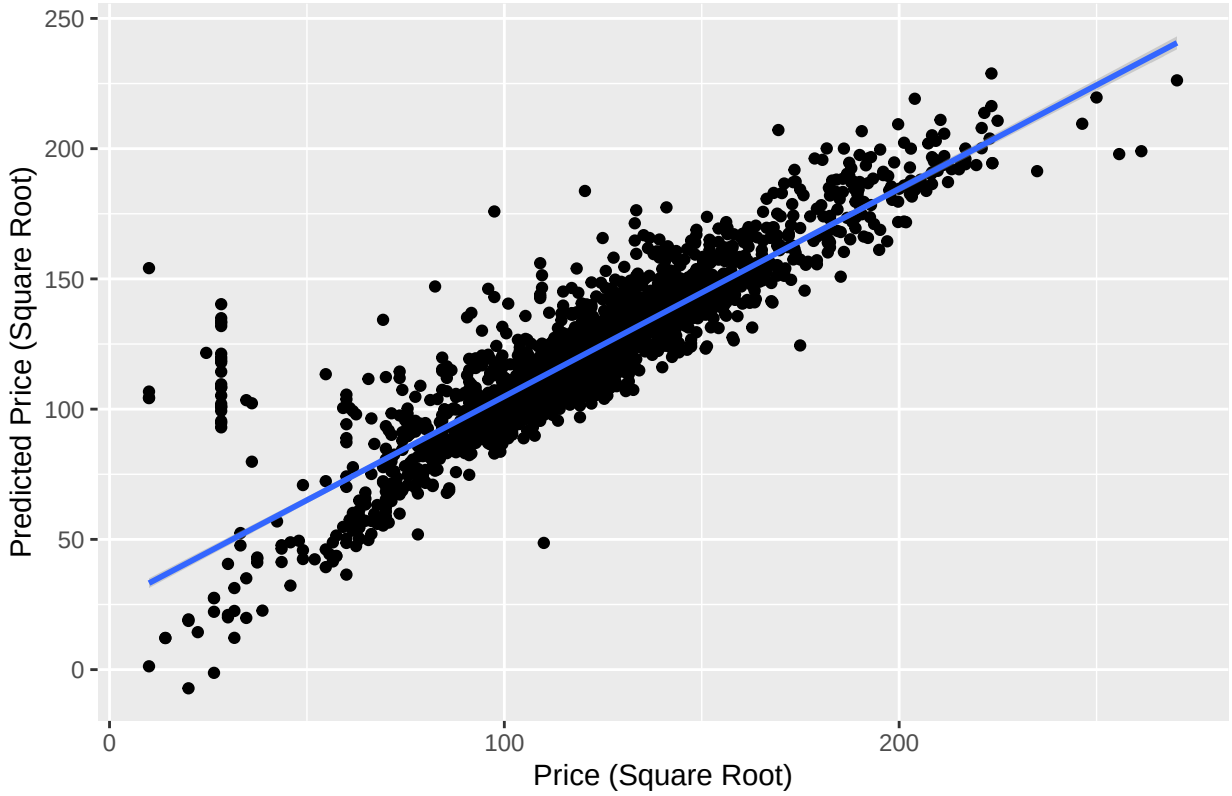
```r
Validation$Predicted_Price_Sqrt_Stepwise1 <- predict(Training_Stepwise_model, newdata = Valida

Validation$Pred_Stepwise1_Original <- Validation$Predicted_Price_Sqrt_Stepwise1 - Validation$p
```

```r
ggplot(Validation, aes(x = price_sqrt, y = Validation$Predicted_Price_Sqrt_Stepwise1)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Figure D1 Scatter Plot of Price (Square Root) vs Predicted Price (Square Root)
       x = "Price (Square Root)",
       y = "Predicted Price (Square Root)")
```

```
## Warning: Use of ‘Validation$Predicted_Price_Sqrt_Stepwise1‘ is discouraged.
## i Use ‘Predicted_Price_Sqrt_Stepwise1‘ instead.
## Use of ‘Validation$Predicted_Price_Sqrt_Stepwise1‘ is discouraged.
## i Use ‘Predicted_Price_Sqrt_Stepwise1‘ instead.
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

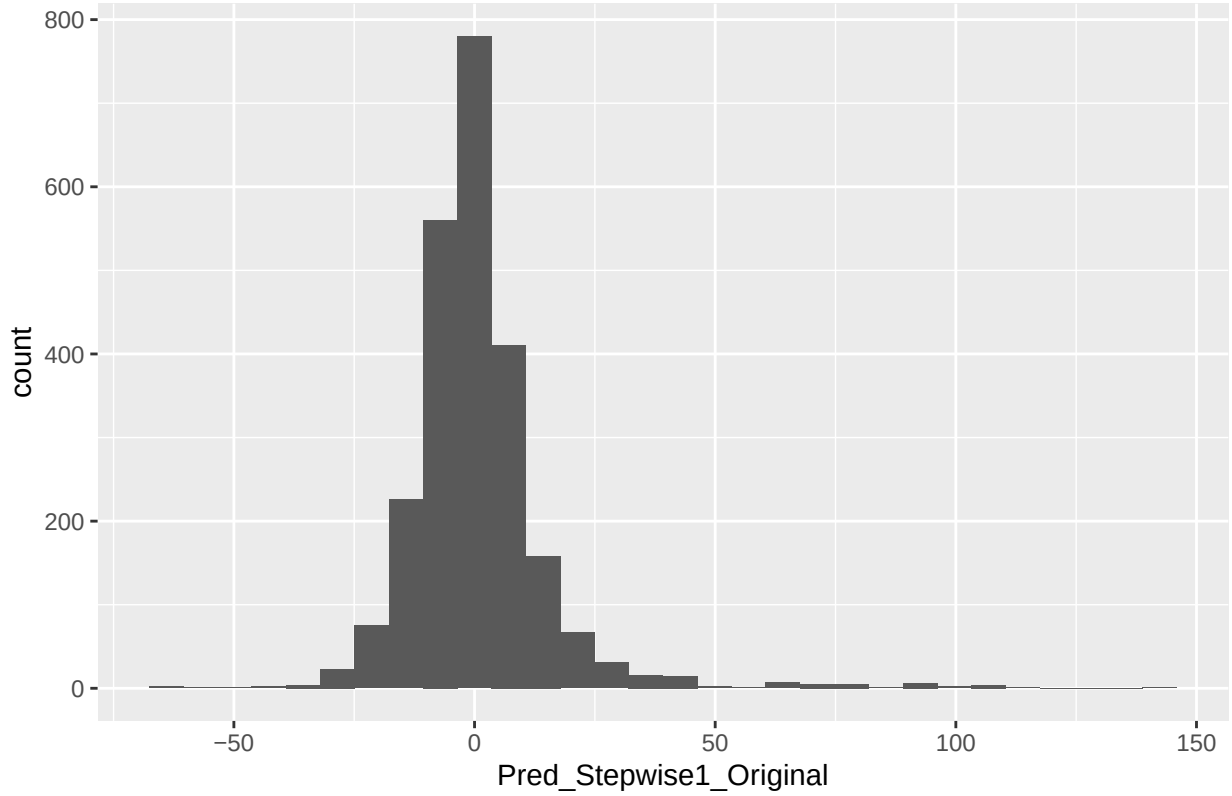
Figure D1 Scatter Plot of Price (Square Root) vs Predicted Price (Square R




```
ggplot(Validation, aes(x = Pred_Stepwise1_Original)) +
  geom_histogram()  +
  labs(title = "Figure D2 Histogram of difference between Predicted Price (Square Root) to Pri
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Figure D2 Histogram of difference between Predicted Price (Square Root) t




```
# Calculate performance metrics: MAE, RMSE, and R-squared
mae <- mean(abs(Validation$Predicted_Price_Sqrt_Stepwise1 - Validation$price_sqrt))
rmse <- sqrt(mean((Validation$Predicted_Price_Sqrt_Stepwise1 - Validation$price_sqrt)^2))

# Calculate R-squared
rss <- sum((Validation$Predicted_Price_Sqrt_Stepwise1 - Validation$price_sqrt)^2)  # Residual
tss <- sum((Validation$price_sqrt - mean(Validation$price_sqrt))^2)       # Total sum of squa
rsq <- 1 - (rss / tss)  # R-squared

# Print performance metrics
cat("Validation Set Performance Metrics for Stepwise Model:\n")
```

```
## Validation Set Performance Metrics for Stepwise Model:
```

```
cat("Mean Absolute Error (MAE):", mae, "\n")
```

```
## Mean Absolute Error (MAE): 9.142246
```

```
cat("Root Mean Squared Error (RMSE):", rmse, "\n")
```

```
## Root Mean Squared Error (RMSE): 15.0786
```

```r
cat("R-squared (R²):", rsq, "\n")
```

```
## R-squared (R²): 0.791915
```