CAR RESALE VALUE PREDICTION USING MULTI-LINEAR REGRESSION

Presented by:
Keerthana Goka
Peiyu Sun
Vinay Konduru
Ashok Jangam

Abstract

With this project, we aim to show how linear regression may be used practically in the automobile sector, by estimating the worth of cars during resale, leading to more informed decisions. By examining variables like brand, engine power, mileage, and year of production, a statistical technique called linear regression is useful for estimating automobile pricing. This approach finds correlations between these attributes and automobile pricing by looking at past sales data. Both buyers and sellers may make more informed selections with the help of this prediction model. Cars offered can be valued by buyers, and sellers can establish competition.

Table of Content

Chapter 1: Proposal ………………………………………………………………………... 5

1.a Introduction

1.b Objective

1.c Research Questions

1.d Hypothesis

1.e Analysis Plan

1.f Data

1.g Initial Correlation Matrix of the dataset

Chapter 2: Library used ……………………………………………………………………… 9

Chapter 3: Data Exploration ……………………………………………………………….. 10

Chapter 4: Creating Quadratic and Interaction terms …………………………………… 14

Chapter 5: Data Splitting ………………………….……………………………………….. 15

Chapter 6: Model Selection ………………………………………………………………. 17

Chapter 7: Multicollinearity Test - VIF – ModelF ………………………………………… 19

Chapter 8: ModelF Selection using Cp, BIC, R2 and Adj R2 …………………………….. 21

Chapter 9: ModelB Selection using Cp, BIC, R2 and Adj R2 ……………………………. 22

Chapter 10: ModelS Selection using Cp, BIC, R2 and Adj R2 …………………………… 23

Chapter 11: Multicollinearity Test - VIF- ModelS ……………………….……………... 24

Chapter 12: Multiple Linear Regression ……………………………………………….….. 25

Chapter 13: Extra Sum of Squares ………………………………………………………… 28

Chapter 14: ModelF/ ModelS Outlier and Regression Analysis ………………………….. 30

Chapter 15: Final MLR ModelF ……………………………………………………………. 35

Chapter 16: Model validation of ModelF ………………………………………………... 37

Chapter 17: Final MLR ModelS …………………………………………………………… 38

Chapter 18: Model Validation of ModelS ……………………………………………….. 39

Chapter 19: Practical Significance of Models and Variables ……………………………... 42

Chapter 20: References ……………………………………………………………………... 43

Proposal

Introduction**:**

The market for second-hand automobiles has experienced significant growth in recent years, enabling buyers and sellers to engage in transactions at fair and acceptable rates. With scientific analysis of car prices, the car market benefits from Fair Pricing, Risk Reduction, Market Efficiency, Consumer Empowerment, Business Optimization, and Data-Driven Insight

Further, Hyundai cars are known for their reliability and durability, consistently ranking well in terms of longevity and performance.

Objective:

The objective of this project is to find the most significant factors to predict Pre-owned Car Prices of Hyundai in India using Multilinear regression techniques.

Research Questions:

1. What are the key factors having a higher correlation to the selling prices of pre-owned cars?
2. How does each factor contribute to the overall variation in car prices?
3. What is the relationship between the independent variables (e.g., mileage, km driven, seats) and the dependent variable (car price)?
4. How well does the multilinear regression model fit the observed data?
5. Are there any outliers or influential data points that significantly impact the model's performance?
6. How can the model be interpreted and applied by stakeholders in the pre-owned car industry?

Hypothesis:

The hypothesis is that all the variables considered in the dataset are highly correlated and have no multicollinearity among them.

Analysis Plan:

The data in consideration has 7 quantitative fields and 4 qualitative fields with Selling price as the Observed variable.

Sno	Variable	Observed/Predictor	Qualitative/Quantitative
1	Selling_Price	Observed Variable	Quantitative variables
2	Year	Predictor Variables	
3	Kms_Driven		
4	Mileage		
5	Engine		
6	Max_Power		
7	Seats		
8	Fuel		Qualitative variables
9	Seller_Type		
10	Transmission		
11	Owner		

- The analysis plan for the **Multilinear regression model** is:
  1. Data Preprocessing: Clean data, handle missing values, and split into training/testing sets.
  2. Exploratory Data Analysis (EDA): Understand variable distributions and relationships.
  3. Model Building: Select predictors, and fit the multilinear regression model.
  4. Model Evaluation: Assess model performance using testing data and metrics like MSE, MAE, and R-squared.
  5. Interpretation: Understand coefficient impact, significance, and practical implications.
  6. Model Improvement: Address multicollinearity, consider feature engineering and try alternative models.
  7. Validation and Sensitivity Analysis: Validate the model with cross-validation, and perform sensitivity analysis.
  8. Documentation and Reporting: Document analysis process, prepare a report summarizing findings and recommendations.

Data:

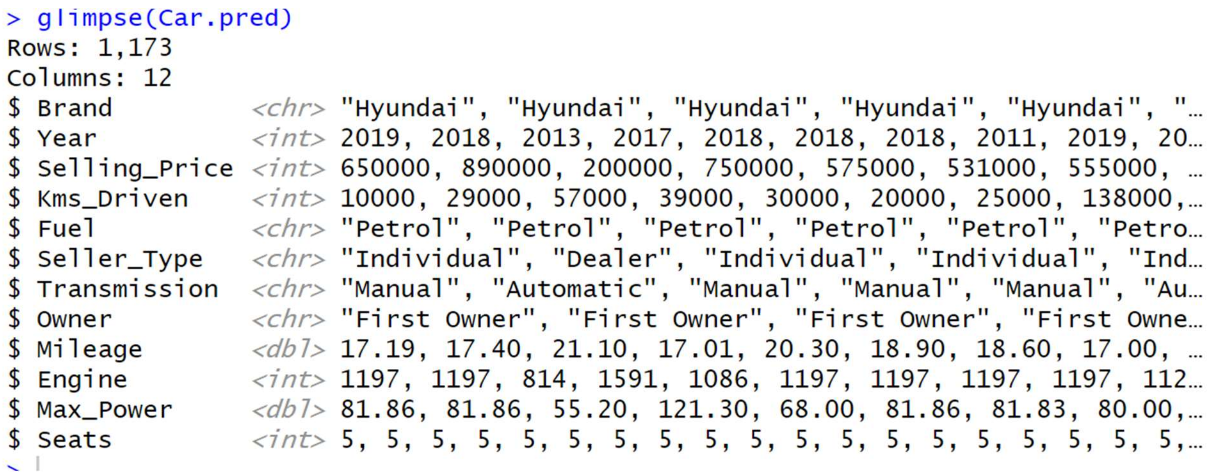
- The considered dataset is taken from Kaggle.com.
- No of observations: 1173
- No of variables: 11

- Data dictionary:

Sn o	Variable	Description
1	Selling_Price	This is the observed variable in multilinear regression analysis. It represents the price at which the car is being sold.
2	Year	This predictor variable represents the year in which the car was manufactured. It could influence the selling price as newer cars tend to have higher prices.
3	Kms_Driven	This predictor variable represents the total distance (in kilometers) the car has been driven. It could affect the selling price, as cars with lower usage are typically perceived to be in better condition and may fetch a higher price.
4	Mileage	This predictor variable represents the fuel efficiency or mileage of the car (measured in kilometers per liter or similar units). It could influence the selling price, as cars with higher mileage may be perceived as less valuable.
5	Engine	This predictor variable likely represents the engine capacity in cc. It could influence the selling price, as cars with larger engines may have higher performance and thus higher prices.
6	Max_Power	This predictor variable represents the maximum power output of the car's engine (e.g., in horsepower or kilowatts). It could influence the selling price, as cars with higher power output may be perceived as more desirable and thus command higher prices.
7	Seats	This predictor variable represents the number of seats(5 or 7) in the car. It could influence the selling price, as cars with more seats may be more practical for families or larger groups and thus may have higher prices.
8	Fuel	This predictor variable represents the type of fuel used by the car (e.g., petrol, diesel, LPG,CNG). It could influence the selling price, as different fuel types may have different costs associated with them and may appeal to different types of buyers.
9	Seller_Type	This predictor variable represents the type of seller (e.g., dealer, individual). It could influence the selling price, as cars sold by dealers may have higher prices due to warranties or other factors.
10	Transmission	This predictor variable represents the type of transmission (e.g., manual, automatic) used by the car. It could influence the selling price, as cars with automatic transmissions may have higher prices due to perceived convenience.

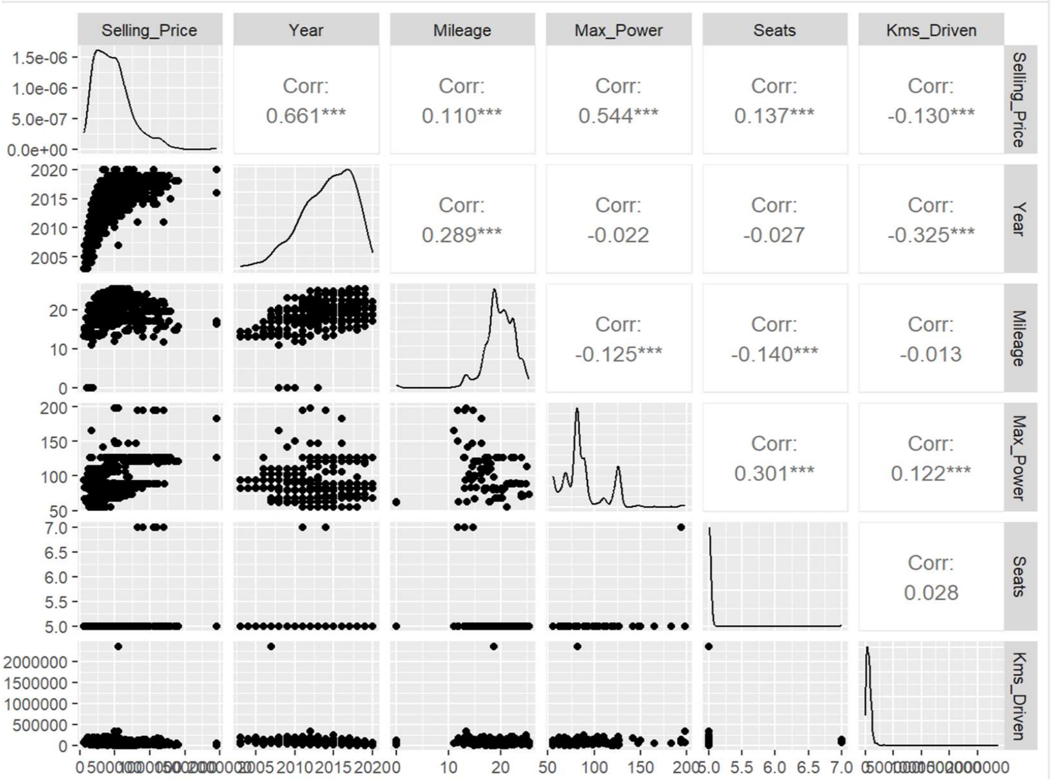
11	Owner	This predictor variable represents the number of previous owners of the car. It could influence the selling price, as cars with fewer previous owners may be perceived as being in better condition and may have higher prices.

- Glimpse of the Dataset:



```
> glimpse(Car.pred)
Rows: 1,173
Columns: 12
$ Brand         <chr> "Hyundai", "Hyundai", "Hyundai", "Hyundai", "Hyundai", "…
$ Year          <int> 2019, 2018, 2013, 2017, 2018, 2018, 2018, 2011, 2019, 20…
$ Selling_Price <int> 650000, 890000, 200000, 750000, 575000, 531000, 555000, …
$ Kms_Driven    <int> 10000, 29000, 57000, 39000, 30000, 20000, 25000, 138000,…
$ Fuel          <chr> "Petrol", "Petrol", "Petrol", "Petrol", "Petrol", "Petro…
$ Seller_Type   <chr> "Individual", "Dealer", "Individual", "Individual", "Ind…
$ Transmission  <chr> "Manual", "Automatic", "Manual", "Manual", "Manual", "Au…
$ Owner         <chr> "First Owner", "First Owner", "First Owner", "First Owne…
$ Mileage       <dbl> 17.19, 17.40, 21.10, 17.01, 20.30, 18.90, 18.60, 17.00, …
$ Engine        <int> 1197, 1197, 814, 1591, 1086, 1197, 1197, 1197, 1197, 112…
$ Max_Power     <dbl> 81.86, 81.86, 55.20, 121.30, 68.00, 81.86, 81.83, 80.00,…
$ Seats         <int> 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,…
```

Initial Correlation Matrix of the dataset:



Library used

In our project, we leverage a diverse set of libraries to enhance our data analysis and visualization capabilities. Two key libraries at the forefront of our toolset are ggplot2 and GGally, which play pivotal roles in crafting insightful visualizations and conducting thorough data analysis. Specifically:

tidyverse: Provides a consistent set of tools for data processing and visualization, including packages like ggplot2 and dplyr, making data processing more concise and standardized.

ggplot2: Used to create various types of statistical charts, it is powerful and flexible, meeting various data visualization needs.

GGally: Offers extended charting and analytical capabilities, especially for multivariate analysis methods like scatterplot matrices.

psych: Provides functions related to psychological statistics, such as factor analysis and descriptive statistics.

ggpubr: Enhances ggplot2 with additional features and styles, allowing for richer and more complex graphics.

leaps: Used for variable selection and model comparison, particularly through forward and backward stepwise selection for model building.

DAAG: Provides tools for data analysis and graphics, particularly useful for educational and experimental design purposes.

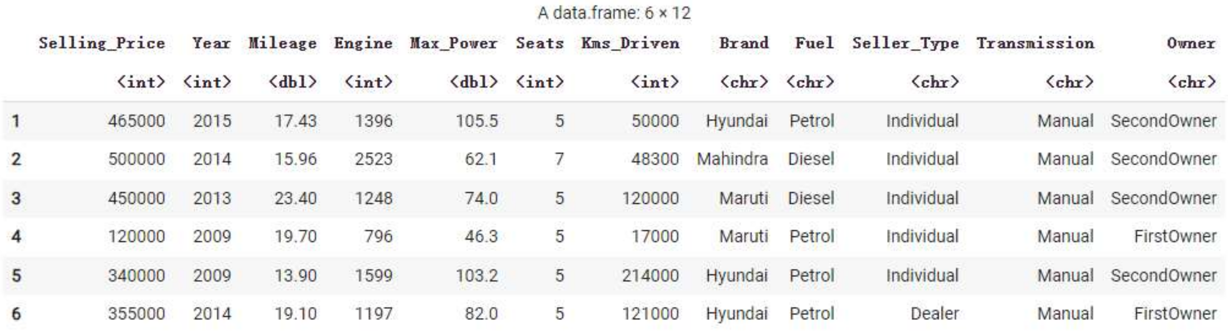
corrplot: Creates correlation matrix plots that visually display the relationships between variables.

olsrr: Offers tools for evaluating and diagnosing linear regression models, including various diagnostic charts and statistical tests.

Data Exploration

In the Data Exploration section, we delved into the initial analysis of our dataset, Car.pred. This involved examining the structure and content of the data to gain a foundational understanding before proceeding with further analysis.

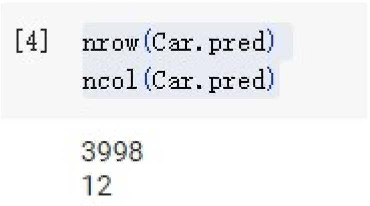
Firstly, we used the head() function to preview the first few rows of the Car.pred dataset, providing an initial glimpse into the data's structure and the type of information it contains.



A data.frame: 6 × 12

	Selling_Price	Year	Mileage	Engine	Max_Power	Seats	Kms_Driven	Brand	Fuel	Seller_Type	Transmission	Owner
	<int>	<int>	<dbl>	<int>	<dbl>	<int>	<int>	<chr>	<chr>	<chr>	<chr>	<chr>
1	465000	2015	17.43	1396	105.5	5	50000	Hyundai	Petrol	Individual	Manual	SecondOwner
2	500000	2014	15.96	2523	62.1	7	48300	Mahindra	Diesel	Individual	Manual	SecondOwner
3	450000	2013	23.40	1248	74.0	5	120000	Maruti	Diesel	Individual	Manual	SecondOwner
4	120000	2009	19.70	796	46.3	5	17000	Maruti	Petrol	Individual	Manual	FirstOwner
5	340000	2009	13.90	1599	103.2	5	214000	Hyundai	Petrol	Individual	Manual	SecondOwner
6	355000	2014	19.10	1197	82.0	5	121000	Hyundai	Petrol	Dealer	Manual	FirstOwner

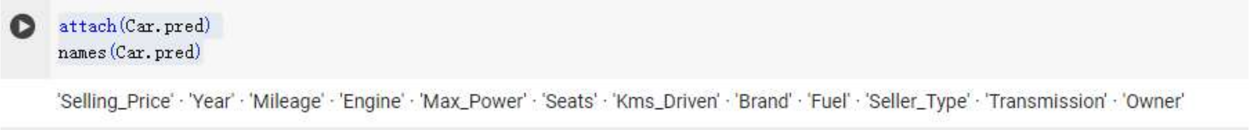
Next, we explored the dimensions of the dataset using nrow() and ncol() functions, which allowed us to understand the number of observations (rows) and variables (columns) present in the dataset, respectively.



```
[4]  nrow(Car.pred)
     ncol(Car.pred)

     3998
     12
```

Additionally, we utilized the attach() function to facilitate easier access to the variables within the dataset, simplifying subsequent data manipulation and analysis tasks.

Lastly, we employed the names() function to retrieve the variable names of the Car.pred dataset, aiding in the identification and understanding of the different attributes captured in the data.



```
attach(Car.pred)
names(Car.pred)
```

```
'Selling_Price' · 'Year' · 'Mileage' · 'Engine' · 'Max_Power' · 'Seats' · 'Kms_Driven' · 'Brand' · 'Fuel' · 'Seller_Type' · 'Transmission' · 'Owner'
```

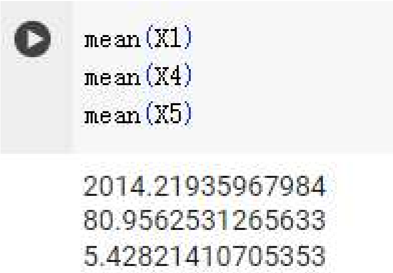
Overall, this section provided a crucial initial exploration of the dataset, laying the groundwork for more in-depth analysis and insights into the data's characteristics and patterns.

Declaring Dependent and Independent Variables

In this section, we engaged in a series of crucial steps to define our dependent and independent variables, ensuring a solid foundation for subsequent analysis and modeling.

Continuous Variables

We initiated by declaring our continuous variables, encompassing attributes such as Year, Mileage, Engine, Max Power, Seats, and Kms Driven. This initial step allowed us to delve into the central tendencies of these variables through mean calculations, providing valuable insights into their distribution within the dataset.



```
mean(X1)
mean(X4)
mean(X5)
```

```
2014.21935967984
80.9562531265633
5.42821410705353
```

Centering Variables around the Means

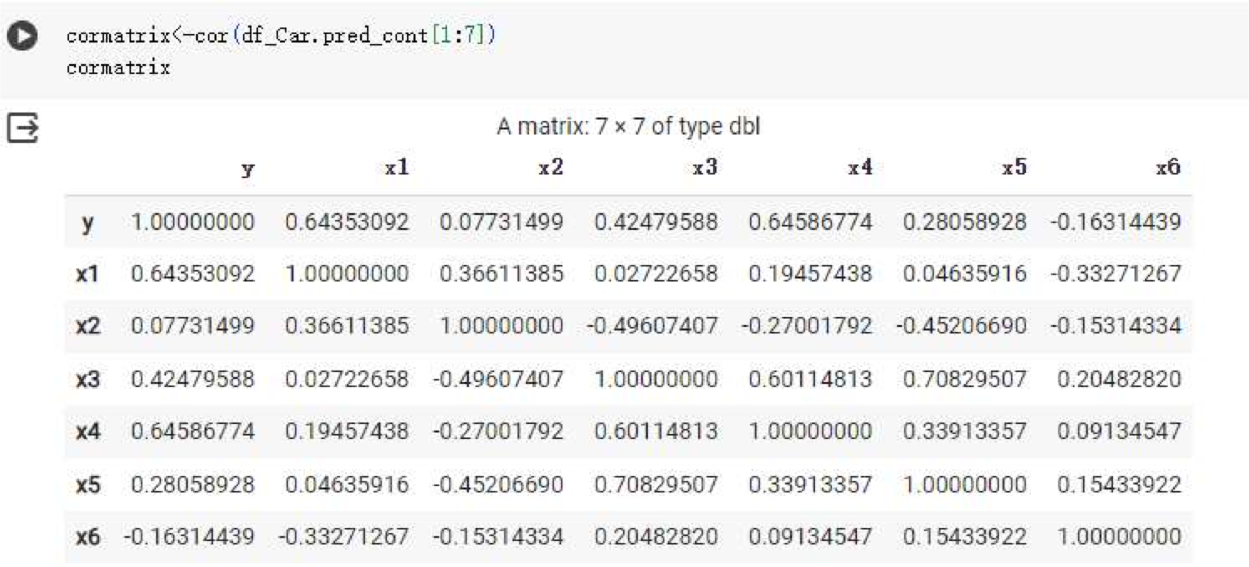
To address multicollinearity concerns effectively, we centered our continuous variables around their respective means. This centering process aids in reducing collinearity among variables, thereby enhancing the reliability of subsequent testing and modeling procedures.

Dependent Variable Transformation

Our focus then shifted to transforming the dependent variable, Selling Price, through a logarithmic base 10 transformation (log10). This transformation is instrumental in addressing skewed distributions, facilitating improved linearity for regression modeling.

Correlation Analysis

Conducting a comprehensive correlation analysis was paramount in understanding the relationships between variables. Notably, we observed strong correlations between Selling Price and certain continuous variables, namely Year, Engine, and Max Power. These findings underscore the potential significance of these variables in our predictive models.



```
cormatrix<-cor(df_Car.pred_cont[1:7])
cormatrix
```

A matrix: 7 × 7 of type dbl

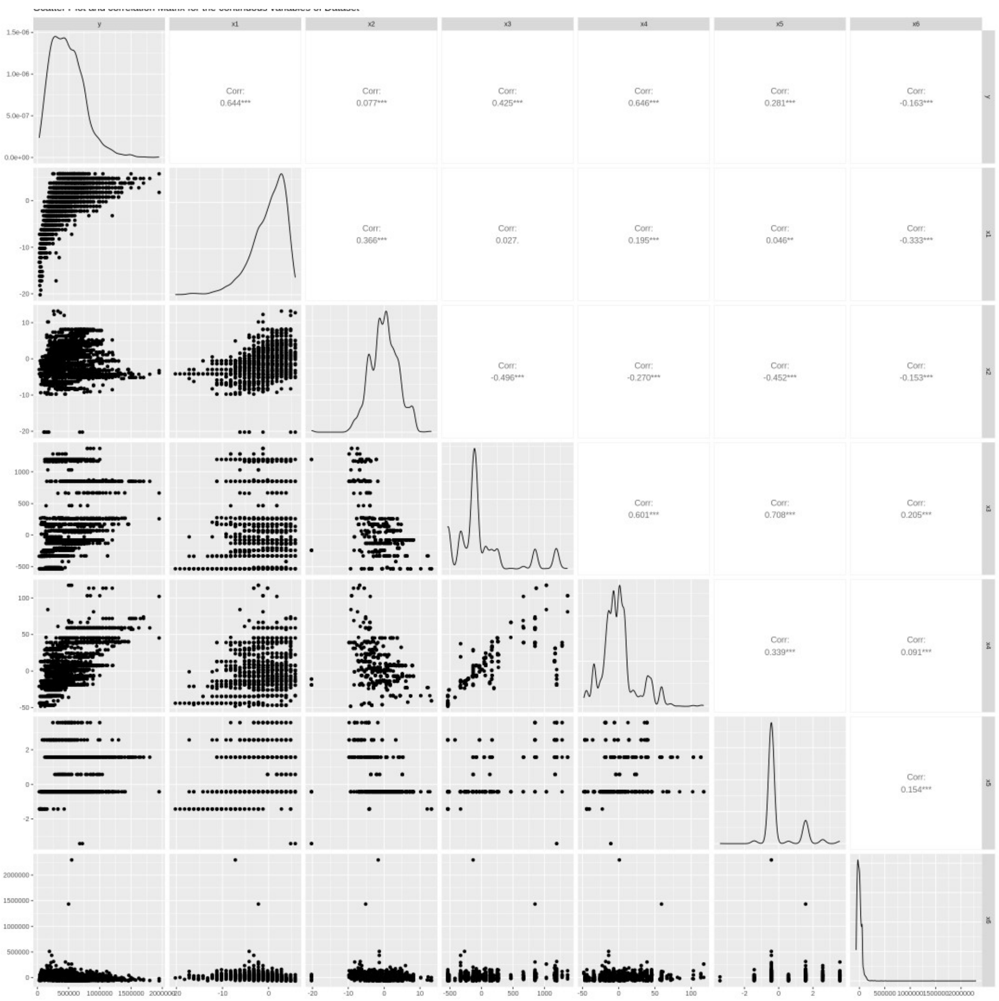
	y	x1	x2	x3	x4	x5	x6
y	1.00000000	0.64353092	0.07731499	0.42479588	0.64586774	0.28058928	-0.16314439
x1	0.64353092	1.00000000	0.36611385	0.02722658	0.19457438	0.04635916	-0.33271267
x2	0.07731499	0.36611385	1.00000000	-0.49607407	-0.27001792	-0.45206690	-0.15314334
x3	0.42479588	0.02722658	-0.49607407	1.00000000	0.60114813	0.70829507	0.20482820
x4	0.64586774	0.19457438	-0.27001792	0.60114813	1.00000000	0.33913357	0.09134547
x5	0.28058928	0.04635916	-0.45206690	0.70829507	0.33913357	1.00000000	0.15433922
x6	-0.16314439	-0.33271267	-0.15314334	0.20482820	0.09134547	0.15433922	1.00000000

Elimination of Highly Correlated Variables

Based on the correlation analysis outcomes, we made strategic decisions to exclude highly correlated variables, such as Engine (X3), due to their potential multicollinearity effects with other variables like Max Power (X4) and Seats (X5). This elimination step aimed to enhance the robustness and interpretability of our models.

Categorical Variables

Our analysis extended to processing categorical variables related to Brand, Fuel type, Seller Type, Transmission, and Owner status. These variables underwent encoding into numerical values, facilitating their inclusion in our analytical framework and contributing to a holistic understanding of the dataset's characteristics.

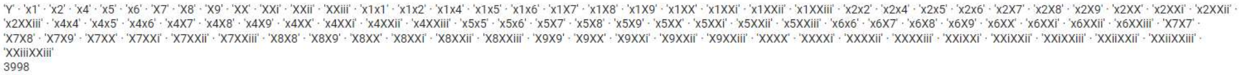


In summary, this section encompassed meticulous data preparation, transformation, and correlation analysis, laying a solid groundwork for subsequent modeling endeavors. The insights gleaned from these analytical steps are pivotal in crafting accurate and reliable predictive models.

Creating Quadratic and Interaction terms

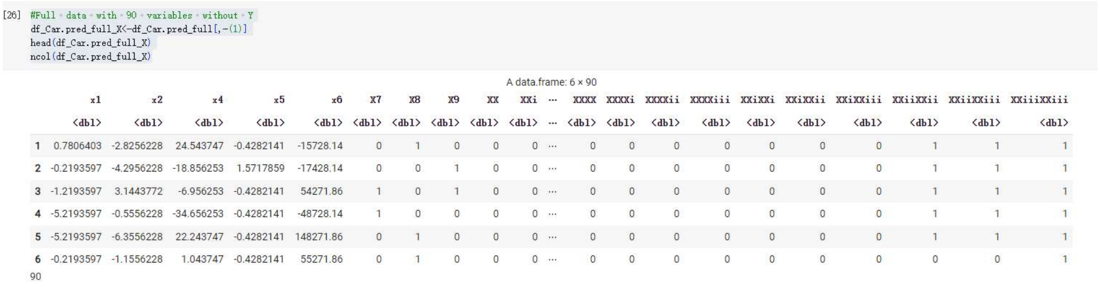
In this section, we extended our dataset by creating quadratic terms and interaction terms for the continuous variables to enhance the complexity and predictive power of our models.

We began by duplicating our dataset df_Car.pred into df_Car.pred_full to maintain the original data while adding new features. Using nested loops, we systematically generated quadratic terms and interaction terms between pairs of continuous variables. This process involved multiplying each pair of variables and appending the resulting product as new columns in df_Car.pred_full.



```
'Y' · 'x1' · 'x2' · 'x4' · 'x5' · 'x6' · 'X7' · 'X8' · 'X9' · 'XX' · 'XXi' · 'XXii' · 'XXiii' · 'x1x1' · 'x1x2' · 'x1x4' · 'x1x5' · 'x1x6' · 'x1X7' · 'x1X8' · 'x1X9' · 'x1XX' · 'x1XXi' · 'x1XXii' · 'x1XXiii' · 'x2x2' · 'x2x4' · 'x2x5' · 'x2x6' · 'x2X7' · 'x2X8' · 'x2X9' · 'x2XX' · 'x2XXi' · 'x2XXii' ·
'x2XXiii' · 'x4x4' · 'x4x5' · 'x4x6' · 'x4X7' · 'x4X8' · 'x4X9' · 'x4XX' · 'x4XXi' · 'x4XXii' · 'x4XXiii' · 'x5x5' · 'x5x6' · 'x5X7' · 'x5X8' · 'x5X9' · 'x5XX' · 'x5XXi' · 'x5XXii' · 'x5XXiii' · 'x6x6' · 'x6X7' · 'x6X8' · 'x6X9' · 'x6XX' · 'x6XXi' · 'x6XXii' · 'x6XXiii' · 'X7X7' ·
'X7X8' · 'X7X9' · 'X7XX' · 'X7XXi' · 'X7XXii' · 'X7XXiii' · 'X8X8' · 'X8X9' · 'X8XX' · 'X8XXi' · 'X8XXii' · 'X8XXiii' · 'X9X9' · 'X9XX' · 'X9XXi' · 'X9XXii' · 'X9XXiii' · 'XXXX' · 'XXXXi' · 'XXXXii' · 'XXXXiii' · 'XXiXXi' · 'XXiXXii' · 'XXiXXiii' · 'XXiiXXii' · 'XXiiXXiii' ·
'XXiiiXXiii'
3998
```

After creating these new variables, we observed that the total number of variables increased to 90, indicating the successful addition of quadratic and interaction terms. The dataset df_Car.pred_full_X was then modified to exclude the response variable, resulting in a matrix df_Car.pred_full_X_M containing the predictor variables along with their quadratic and interaction terms.



```
[26] #Full data with 90 variables without Y
     df_Car.pred_full_X<-df_Car.pred_full[,-(1)]
     head(df_Car.pred_full_X)
     ncol(df_Car.pred_full_X)
```

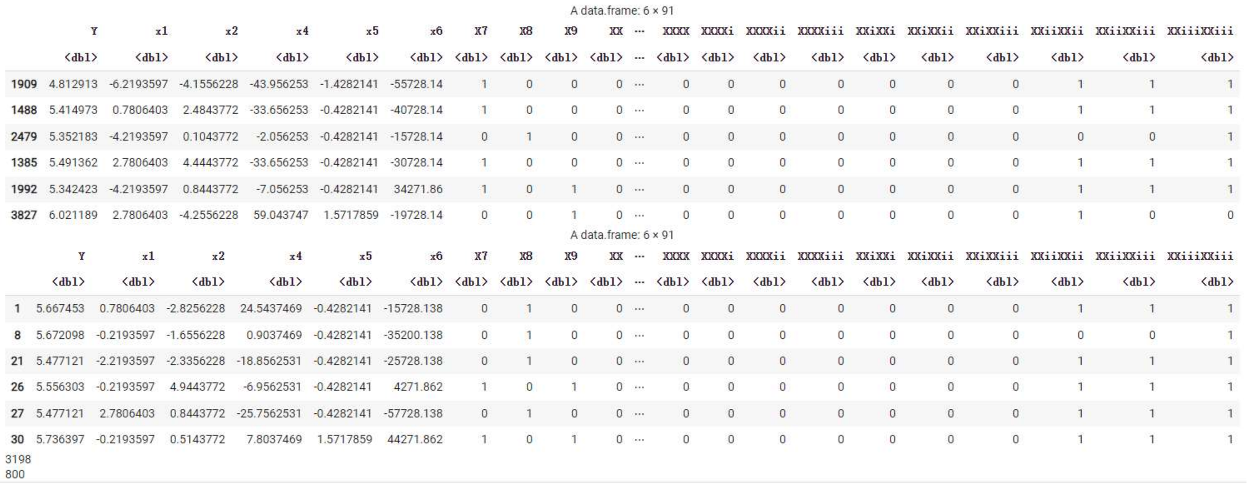
A data.frame: 6 × 90

	x1	x2	x4	x5	x6	X7	X8	X9	XX	XXi	⋯	XXXX	XXXXi	XXXXii	XXXXiii	XXiXXi	XXiXXii	XXiXXiii	XXiiXXii	XXiiXXiii	XXiiiXXiii
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.7806403	-2.8256228	24.543747	-0.4282141	-15728.14	0	1	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
2	-0.2193597	-4.2956228	-18.856253	1.5717859	-17428.14	0	0	1	0	0	⋯	0	0	0	0	0	0	0	1	1	1
3	-1.2193597	3.1443772	-6.956253	-0.4282141	54271.86	1	0	1	0	0	⋯	0	0	0	0	0	0	0	1	1	1
4	-5.2193597	-0.5556228	-34.656253	-0.4282141	-48728.14	1	0	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
5	-5.2193597	-6.3556228	22.243747	-0.4282141	148271.86	0	1	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
6	-0.2193597	-1.1556228	1.043747	-0.4282141	55271.86	0	1	0	0	0	⋯	0	0	0	0	0	0	0	0	0	1

90

Data Splitting

In this section, we conducted data splitting to create a reproducible sample for training and testing our models. Specifically, 80% of the dataset was allocated for training (Car.pred_train), while the remaining 20% was reserved for validation (Car.pred_validation). This partitioning was achieved using random sampling with a seed value (2023) to ensure reproducibility.



A data.frame: 6 × 91

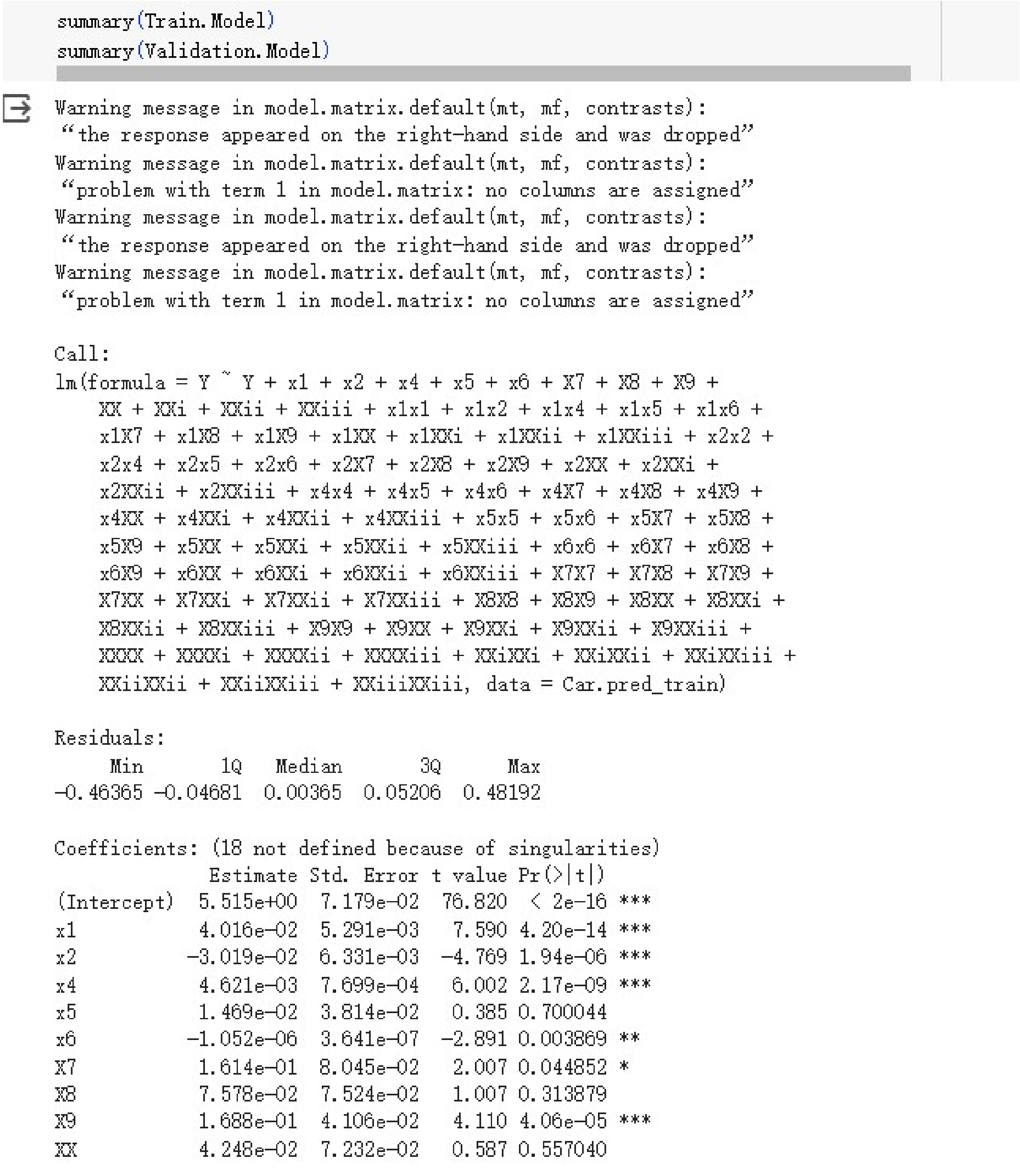
	Y	x1	x2	x4	x5	x6	X7	X8	X9	XX	⋯	XXXX	XXXXi	XXXXii	XXXXiii	XXiXXi	XXiXXii	XXiXXiii	XXiiXXii	XXiiXXiii	XXiiiXXiii
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-4.1556228	-43.956253	-1.4282141	-55728.14	1	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
1488	5.414973	0.7806403	2.4843772	-33.656253	-0.4282141	-40728.14	1	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
2479	5.352183	-4.2193597	0.1043772	-2.056253	-0.4282141	-15728.14	0	1	0	0	⋯	0	0	0	0	0	0	0	0	0	1
1385	5.491362	2.7806403	4.4443772	-33.656253	-0.4282141	-30728.14	1	0	0	0	⋯	0	0	0	0	0	0	0	1	1	1
1992	5.342423	-4.2193597	0.8443772	-7.056253	-0.4282141	34271.86	1	0	1	0	⋯	0	0	0	0	0	0	0	1	1	1
3827	6.021189	2.7806403	-4.2556228	59.043747	1.5717859	-19728.14	0	0	1	0	⋯	0	0	0	0	0	0	0	1	0	0

A data.frame: 6 × 91

	Y	x1	x2	x4	x5	x6	X7	X8	X9	XX	⋯	XXXX	XXXXi	XXXXii	XXXXiii	XXiXXi	XXiXXii	XXiXXiii	XXiiXXii	XXiiXXiii	XXiiiXXiii
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	5.667453	0.7806403	-2.8256228	24.5437469	-0.4282141	-15728.138	0	1	0	0	⋯	0	0	0	0	0	0	0	1	1	1
8	5.672098	-0.2193597	-1.6556228	0.9037469	-0.4282141	-35200.138	0	1	0	0	⋯	0	0	0	0	0	0	0	0	0	1
21	5.477121	-2.2193597	-2.3356228	-18.8562531	-0.4282141	-25728.138	0	1	0	0	⋯	0	0	0	0	0	0	0	1	1	1
26	5.556303	-0.2193597	4.9443772	-6.9562531	-0.4282141	4271.862	1	0	1	0	⋯	0	0	0	0	0	0	0	1	1	1
27	5.477121	2.7806403	0.8443772	-25.7562531	-0.4282141	-57728.138	0	1	0	0	⋯	0	0	0	0	0	0	0	1	1	1
30	5.736397	-0.2193597	0.5143772	7.8037469	1.5717859	44271.862	1	0	1	0	⋯	0	0	0	0	0	0	0	1	1	1

```
3198
800
```

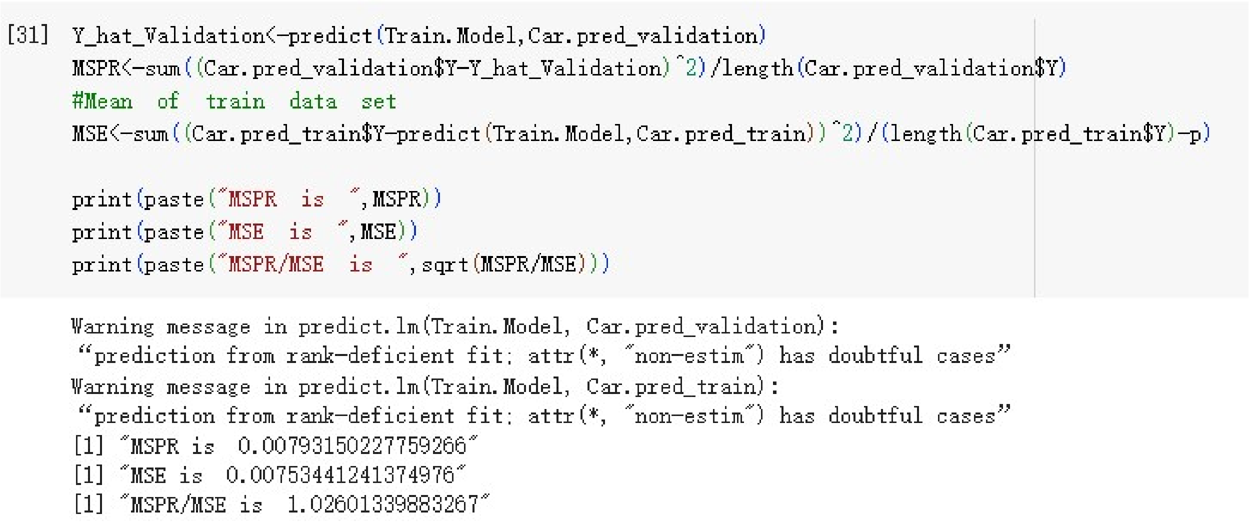
After splitting the data, we constructed linear regression models for both the training and validation datasets (Train.Model and Validation.Model, respectively). These models were built using predictor variables, including their quadratic and interaction terms, to capture complex relationships and interactions among variables.



```
summary(Train.Model)
summary(Validation.Model)
```

```
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"
Warning message in model.matrix.default(mt, mf, contrasts):
"the response appeared on the right-hand side and was dropped"
Warning message in model.matrix.default(mt, mf, contrasts):
"problem with term 1 in model.matrix: no columns are assigned"

Call:
lm(formula = Y ~ Y + x1 + x2 + x4 + x5 + x6 + X7 + X8 + X9 +
    XX + XXi + XXii + XXiii + x1x1 + x1x2 + x1x4 + x1x5 + x1x6 +
    x1X7 + x1X8 + x1X9 + x1XX + x1XXi + x1XXii + x1XXiii + x2x2 +
    x2x4 + x2x5 + x2x6 + x2X7 + x2X8 + x2X9 + x2XX + x2XXi +
    x2XXii + x2XXiii + x4x4 + x4x5 + x4x6 + x4X7 + x4X8 + x4X9 +
    x4XX + x4XXi + x4XXii + x4XXiii + x5x5 + x5x6 + x5X7 + x5X8 +
    x5X9 + x5XX + x5XXi + x5XXii + x5XXiii + x6x6 + x6X7 + x6X8 +
    x6X9 + x6XX + x6XXi + x6XXii + x6XXiii + X7X7 + X7X8 + X7X9 +
    X7XX + X7XXi + X7XXii + X7XXiii + X8X8 + X8X9 + X8XX + X8XXi +
    X8XXii + X8XXiii + X9X9 + X9XX + X9XXi + X9XXii + X9XXiii +
    XXXX + XXXXi + XXXXii + XXXXiii + XXiXXi + XXiXXii + XXiXXiii +
    XXiiXXii + XXiiXXiii + XXiiiXXiii, data = Car.pred_train)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.46365 -0.04681  0.00365  0.05206  0.48192 

Coefficients: (18 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  5.515e+00  7.179e-02  76.820  < 2e-16 ***
x1           4.016e-02  5.291e-03   7.590 4.20e-14 ***
x2          -3.019e-02  6.331e-03  -4.769 1.94e-06 ***
x4           4.621e-03  7.699e-04   6.002 2.17e-09 ***
x5           1.469e-02  3.814e-02   0.385 0.700044    
x6          -1.052e-06  3.641e-07  -2.891 0.003869 ** 
X7           1.614e-01  8.045e-02   2.007 0.044852 *  
X8           7.578e-02  7.524e-02   1.007 0.313879    
X9           1.688e-01  4.106e-02   4.110 4.06e-05 ***
XX           4.248e-02  7.232e-02   0.587 0.557040    
```

Additionally, we evaluated the predictive performance of the models using Mean Squared Prediction Error (MSPR) and Mean Squared Error (MSE).



```
[31] Y_hat_Validation<-predict(Train.Model,Car.pred_validation)
     MSPR<-sum((Car.pred_validation$Y-Y_hat_Validation)^2)/length(Car.pred_validation$Y)
     #Mean  of  train  data  set
     MSE<-sum((Car.pred_train$Y-predict(Train.Model,Car.pred_train))^2)/(length(Car.pred_train$Y)-p)

     print(paste("MSPR  is  ",MSPR))
     print(paste("MSE  is  ",MSE))
     print(paste("MSPR/MSE  is  ",sqrt(MSPR/MSE)))

     Warning message in predict.lm(Train.Model, Car.pred_validation):
     "prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases"
     Warning message in predict.lm(Train.Model, Car.pred_train):
     "prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases"
     [1] "MSPR is  0.00793150227759266"
     [1] "MSE is  0.00753441241374976"
     [1] "MSPR/MSE is  1.02601339883267"
```

The comparison revealed that MSPR was 2% higher than MSE, indicating that MSE is unbiased and demonstrates excellent prediction ability of the train model.

Model Selection

In the Model Selection section, we conducted various analyses to choose the most suitable model for our data. Here's a subjective description of what we did and the methods we used:

Forward Selection

We used the forward selection method to iteratively add variables to the model that improve its predictive power. The output model includes a set of variables that collectively enhance the model's ability to predict outcomes.

Backward Elimination

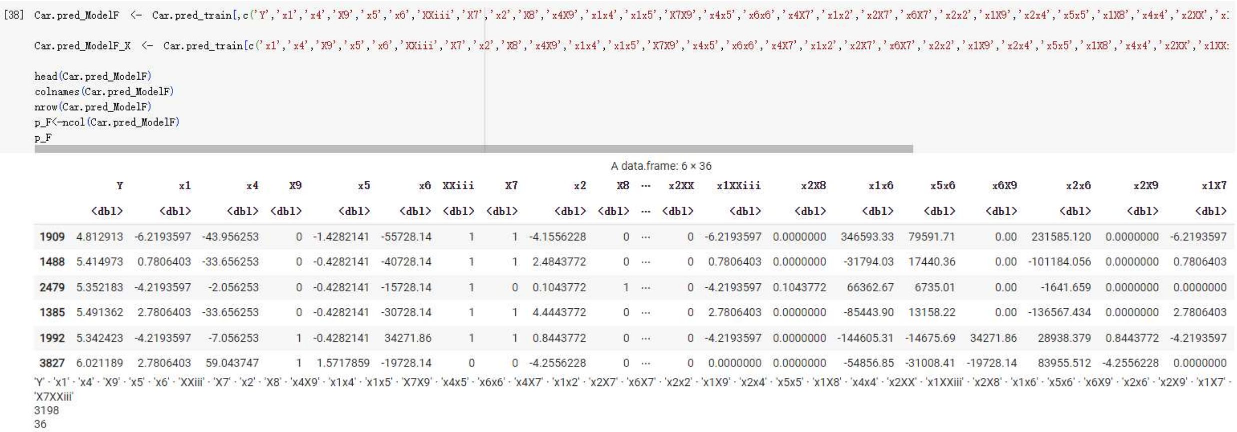
Through backward elimination, we systematically removed variables from the model that did not significantly contribute to its predictive ability. The resulting model is streamlined and includes only the most influential variables.

Stepwise Selection

Using stepwise selection, we combined elements of forward and backward selection to refine the model further. This method allows for a more comprehensive evaluation of variable importance, resulting in a model with optimal predictive power.

After applying the model selection techniques, we obtained three distinct models: ModelF from forward selection, ModelB from backward elimination, and ModelS from stepwise selection. Each model includes a subset of variables that best explain the variation in the target variable.

ModelF - Forward selection result: x1 + x4 + X9 + x5 + x6 + XXiii + X7 + x2 + X8 + x4X9 + x1x4 + x1x5 + X7X9 + x4x5 + x6x6 + x4X7 + x1x2 + x2X7 + x6X7 + x2x2 + x1X9 + x2x4 + x5x5 + x1X8 + x4x4 + x2XX + x1XXiii + x2X8 + x1x6 + x5x6 + x6X9 + x2x6 + x2X9 + x1X7 + X7XXiii



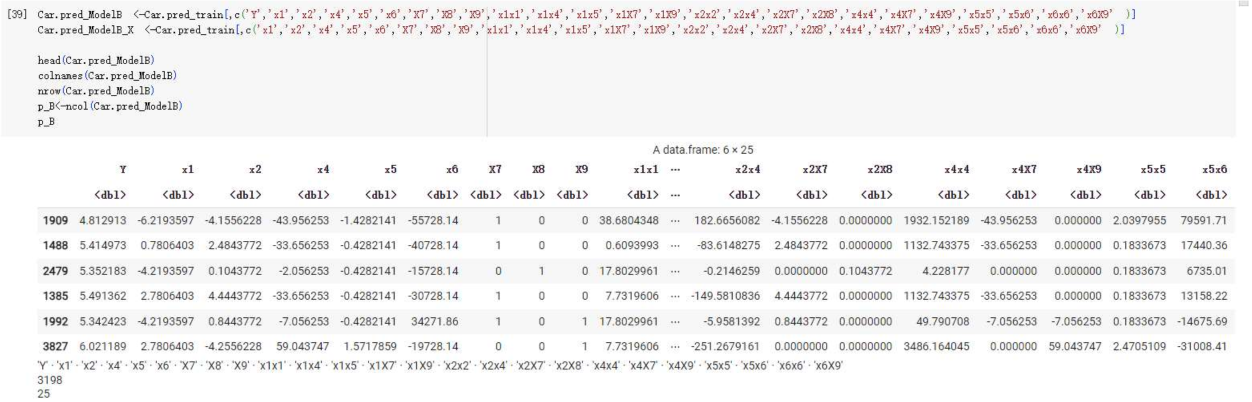
```
[38] Car.pred_ModelF  <-  Car.pred_train[,c('Y','x1','x4','X9','x5','x6','XXiii','X7','x2','X8','x4X9','x1x4','x1x5','X7X9','x4x5','x6x6','x4X7','x1x2','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x1X8','x4x4','x2XX','x

     Car.pred_ModelF_X  <-  Car.pred_train[c('x1','x4','X9','x5','x6','XXiii','X7','x2','X8','x4X9','x1x4','x1x5','X7X9','x4x5','x6x6','x4X7','x1x2','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x1X8','x4x4','x2XX','x1XX

     head(Car.pred_ModelF)
     colnames(Car.pred_ModelF)
     nrow(Car.pred_ModelF)
     p_F<-ncol(Car.pred_ModelF)
     p_F
```

A data.frame: 6 × 36

	Y	x1	x4	X9	x5	x6	XXiii	X7	x2	X8	⋯	x2XX	x1XXiii	x2X8	x1x6	x5x6	x6X9	x2x6	x2X9	x1X7
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-43.956253	0	-1.4282141	-55728.14	1	1	-4.1556228	0	⋯	0	-6.2193597	0.0000000	346593.33	79591.71	0.00	231585.120	0.0000000	-6.2193597
1488	5.414973	0.7806403	-33.656253	0	-0.4282141	-40728.14	1	1	2.4843772	0	⋯	0	0.7806403	0.0000000	-31794.03	17440.36	0.00	-101184.056	0.0000000	0.7806403
2479	5.352183	-4.2193597	-2.056253	0	-0.4282141	-15728.14	1	0	0.1043772	1	⋯	0	-4.2193597	0.1043772	66362.67	6735.01	0.00	-1641.659	0.0000000	0.0000000
1385	5.491362	2.7806403	-33.656253	0	-0.4282141	-30728.14	1	1	4.4443772	0	⋯	0	2.7806403	0.0000000	-85443.90	13158.22	0.00	-136567.434	0.0000000	2.7806403
1992	5.342423	-4.2193597	-7.056253	1	-0.4282141	34271.86	1	1	0.8443772	0	⋯	0	-4.2193597	0.0000000	-144605.31	-14675.69	34271.86	28938.379	0.8443772	-4.2193597
3827	6.021189	2.7806403	59.043747	1	1.5717859	-19728.14	0	0	-4.2556228	0	⋯	0	0.0000000	0.0000000	-54856.85	-31008.41	-19728.14	83955.512	-4.2556228	0.0000000

```
'Y' · 'x1' · 'x4' · 'X9' · 'x5' · 'x6' · 'XXiii' · 'X7' · 'x2' · 'X8' · 'x4X9' · 'x1x4' · 'x1x5' · 'X7X9' · 'x4x5' · 'x6x6' · 'x4X7' · 'x1x2' · 'x2X7' · 'x6X7' · 'x2x2' · 'x1X9' · 'x2x4' · 'x5x5' · 'x1X8' · 'x4x4' · 'x2XX' · 'x1XXiii' · 'x2X8' · 'x1x6' · 'x5x6' · 'x6X9' · 'x2x6' · 'x2X9' · 'x1X7' ·
'X7XXiii'
3198
36
```

ModelB - Backward Elimination result: x1 + x2 + x4 + x5 + x6 + X7 + X8 + X9 + x1x1 + x1x4 + x1x5 + x1X7 + x1X9 + x2x2 + x2x4 + x2X7 + x2X8 + x4x4 + x4X7 + x4X9 + x5x5 + x5x6 + x6x6 + x6X9



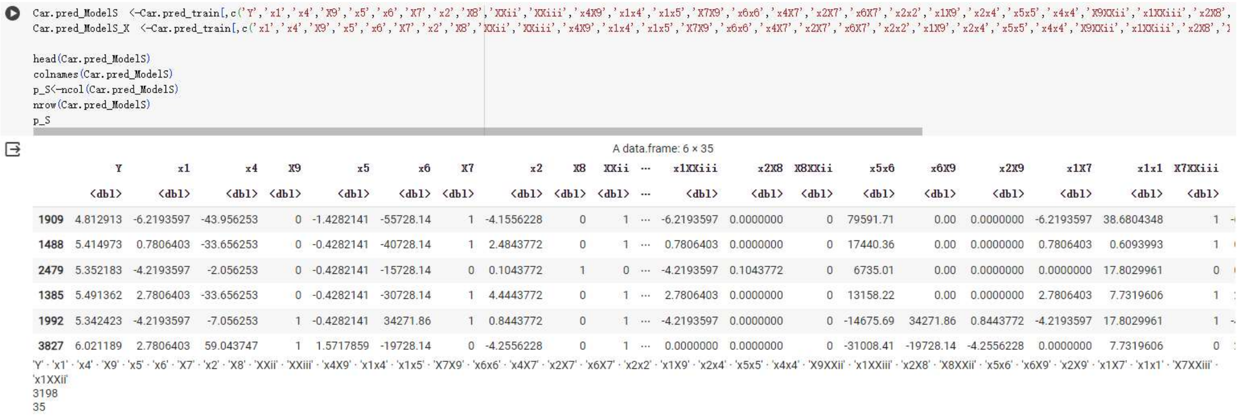
```
[39] Car.pred_ModelB  <-Car.pred_train[,c('Y','x1','x2','x4','x5','x6','X7','X8','X9','x1x1','x1x4','x1x5','x1X7','x1X9','x2x2','x2x4','x2X7','x2X8','x4x4','x4X7','x4X9','x5x5','x5x6','x6x6','x6X9'  )]
     Car.pred_ModelB_X  <-Car.pred_train[,c('x1','x2','x4','x5','x6','X7','X8','X9','x1x1','x1x4','x1x5','x1X7','x1X9','x2x2','x2x4','x2X7','x2X8','x4x4','x4X7','x4X9','x5x5','x5x6','x6x6','x6X9'  )]

     head(Car.pred_ModelB)
     colnames(Car.pred_ModelB)
     nrow(Car.pred_ModelB)
     p_B<-ncol(Car.pred_ModelB)
     p_B
```

A data.frame: 6 × 25

	Y	x1	x2	x4	x5	x6	X7	X8	X9	x1x1	⋯	x2x4	x2X7	x2X8	x4x4	x4X7	x4X9	x5x5	x5x6
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-4.1556228	-43.956253	-1.4282141	-55728.14	1	0	0	38.6804348	⋯	182.6656082	-4.1556228	0.0000000	1932.152189	-43.956253	0.000000	2.0397955	79591.71
1488	5.414973	0.7806403	2.4843772	-33.656253	-0.4282141	-40728.14	1	0	0	0.6093993	⋯	-83.6148275	2.4843772	0.0000000	1132.743375	-33.656253	0.000000	0.1833673	17440.36
2479	5.352183	-4.2193597	0.1043772	-2.056253	-0.4282141	-15728.14	0	1	0	17.8029961	⋯	-0.2146259	0.0000000	0.1043772	4.228177	0.000000	0.000000	0.1833673	6735.01
1385	5.491362	2.7806403	4.4443772	-33.656253	-0.4282141	-30728.14	1	0	0	7.7319606	⋯	-149.5810836	4.4443772	0.0000000	1132.743375	-33.656253	0.000000	0.1833673	13158.22
1992	5.342423	-4.2193597	0.8443772	-7.056253	-0.4282141	34271.86	1	0	1	17.8029961	⋯	-5.9581392	0.8443772	0.0000000	49.790708	-7.056253	-7.056253	0.1833673	-14675.69
3827	6.021189	2.7806403	-4.2556228	59.043747	1.5717859	-19728.14	0	0	1	7.7319606	⋯	-251.2679161	0.0000000	0.0000000	3486.164045	0.000000	59.043747	2.4705109	-31008.41

```
'Y' · 'x1' · 'x2' · 'x4' · 'x5' · 'x6' · 'X7' · 'X8' · 'X9' · 'x1x1' · 'x1x4' · 'x1x5' · 'x1X7' · 'x1X9' · 'x2x2' · 'x2x4' · 'x2X7' · 'x2X8' · 'x4x4' · 'x4X7' · 'x4X9' · 'x5x5' · 'x5x6' · 'x6x6' · 'x6X9'
3198
25
```

ModelS - Stepwise Output model: x1 + x4 + X9 + x5 + x6 + X7 + x2 + X8 +XXii + XXiii + x4X9 + x1x4 + x1x5 + X7X9 + x6x6 + x4X7 + x2X7 + x6X7 + x2x2 + x1X9 + x2x4 + x5x5 + x4x4 + X9XXii + x1XXiii + x2X8 + X8XXii + x5x6 + x6X9 + x2X9 + x1X7 + x1x1 + X7XXiii + x1XXii



```
Car.pred_ModelS  <-Car.pred_train[,c('Y','x1','x4','X9','x5','x6','X7','x2','X8','XXii','XXiii','x4X9','x1x4','x1x5','X7X9','x6x6','x4X7','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x4x4','X9XXii','x1XXiii','x2X8',
Car.pred_ModelS_X  <-Car.pred_train[,c('x1','x4','X9','x5','x6','X7','x2','X8','XXii','XXiii','x4X9','x1x4','x1x5','X7X9','x6x6','x4X7','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x4x4','X9XXii','x1XXiii','x2X8','

head(Car.pred_ModelS)
colnames(Car.pred_ModelS)
p_S<-ncol(Car.pred_ModelS)
nrow(Car.pred_ModelS)
p_S
```

A data.frame: 6 × 35

	Y	x1	x4	X9	x5	x6	X7	x2	X8	XXii	⋯	x1XXiii	x2X8	X8XXii	x5x6	x6X9	x2X9	x1X7	x1x1	X7XXiii
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	⋯	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-43.956253	0	-1.4282141	-55728.14	1	-4.1556228	0	1	⋯	-6.2193597	0.0000000	0	79591.71	0.00	0.0000000	-6.2193597	38.6804348	1
1488	5.414973	0.7806403	-33.656253	0	-0.4282141	-40728.14	1	2.4843772	0	1	⋯	0.7806403	0.0000000	0	17440.36	0.00	0.0000000	0.7806403	0.6093993	1
2479	5.352183	-4.2193597	-2.056253	0	-0.4282141	-15728.14	0	0.1043772	1	0	⋯	-4.2193597	0.1043772	0	6735.01	0.00	0.0000000	0.0000000	17.8029961	0
1385	5.491362	2.7806403	-33.656253	0	-0.4282141	-30728.14	1	4.4443772	0	1	⋯	2.7806403	0.0000000	0	13158.22	0.00	0.0000000	2.7806403	7.7319606	1
1992	5.342423	-4.2193597	-7.056253	1	-0.4282141	34271.86	1	0.8443772	0	1	⋯	-4.2193597	0.0000000	0	-14675.69	34271.86	0.8443772	-4.2193597	17.8029961	1
3827	6.021189	2.7806403	59.043747	1	1.5717859	-19728.14	0	-4.2556228	0	1	⋯	0.0000000	0.0000000	0	-31008.41	-19728.14	-4.2556228	0.0000000	7.7319606	0

```
'Y' · 'x1' · 'x4' · 'X9' · 'x5' · 'x6' · 'X7' · 'x2' · 'X8' · 'XXii' · 'XXiii' · 'x4X9' · 'x1x4' · 'x1x5' · 'X7X9' · 'x6x6' · 'x4X7' · 'x2X7' · 'x6X7' · 'x2x2' · 'x1X9' · 'x2x4' · 'x5x5' · 'x4x4' · 'X9XXii' · 'x1XXiii' · 'x2X8' · 'X8XXii' · 'x5x6' · 'x6X9' · 'x2X9' · 'x1X7' · 'x1x1' · 'X7XXiii' ·
'x1XXii'
3198
35
```

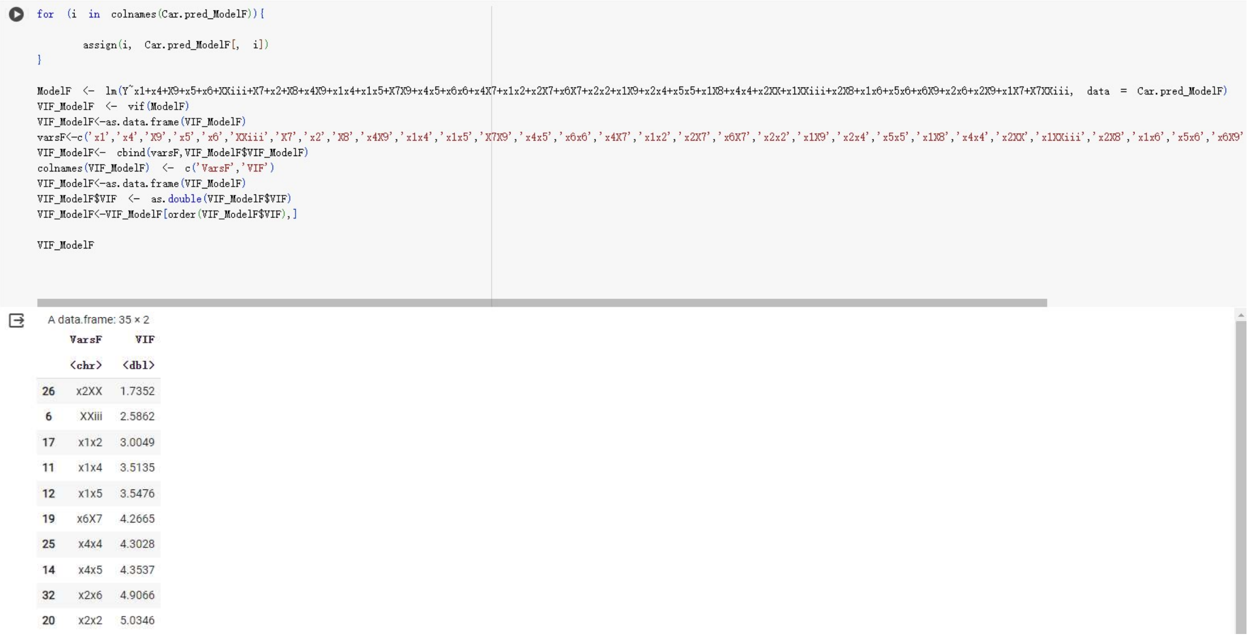
Multicollinearity Test - VIF – ModelF

In this section, we delve into assessing multicollinearity using the Variance Inflation Factor (VIF) and refining the ModelF regression model.

ModelF - VIF Test - Iteration 1

The initial iteration focused on evaluating the VIF for each predictor variable in ModelF. Despite observing the highest VIF for variable x1, its removal was deemed inappropriate due to its strong correlation with the response variable Y. Thus, x1 remained integral to the model, maintaining its significant predictive power.

Following this assessment, the updated ModelF_x1 regression model was constructed without x1, leading to a marginal increase in the Multiple R-squared value, affirming the significance of retaining x1 within the model.

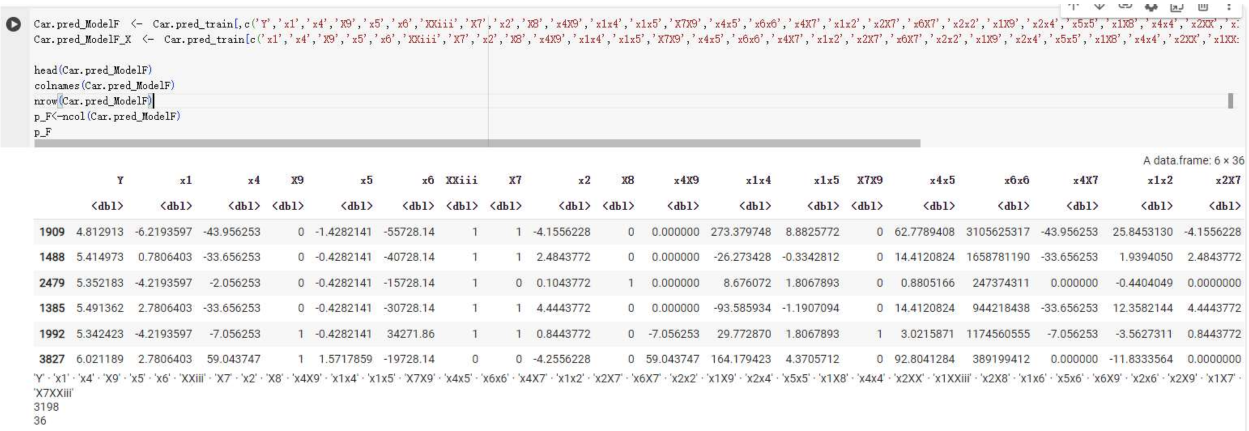


```
for (i in colnames(Car.pred_ModelF)){

      assign(i, Car.pred_ModelF[, i])
}

ModelF <- lm(Y~x1+x4+X9+x5+x6+XXiii+X7+x2+X8+x4X9+x1x4+x1x5+X7X9+x4x5+x6x6+x4X7+x1x2+x2X7+x6X7+x2x2+x1X9+x2x4+x5x5+x1X8+x4x4+x2XX+x1XXiii+x2X8+x1x6+x5x6+x6X9+x2x6+x2X9+x1X7+X7XXiii, data = Car.pred_ModelF)
VIF_ModelF <- vif(ModelF)
VIF_ModelF<-as.data.frame(VIF_ModelF)
varsF<-c('x1','x4','X9','x5','x6','XXiii','X7','x2','X8','x4X9','x1x4','x1x5','X7X9','x4x5','x6x6','x4X7','x1x2','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x1X8','x4x4','x2XX','x1XXiii','x2X8','x1x6','x5x6','x6X9'
VIF_ModelF<- cbind(varsF,VIF_ModelF$VIF_ModelF)
colnames(VIF_ModelF) <- c('VarsF','VIF')
VIF_ModelF<-as.data.frame(VIF_ModelF)
VIF_ModelF$VIF <- as.double(VIF_ModelF$VIF)
VIF_ModelF<-VIF_ModelF[order(VIF_ModelF$VIF),]

VIF_ModelF
```

```
A data.frame: 35 × 2
     VarsF   VIF
     <chr>   <dbl>
26   x2XX    1.7352
6    XXiii   2.5862
17   x1x2    3.0049
11   x1x4    3.5135
12   x1x5    3.5476
19   x6X7    4.2665
25   x4x4    4.3028
14   x4x5    4.3537
32   x2x6    4.9066
20   x2x2    5.0346
```

ModelF After Reduction of Multicollinearity Iterations

The subsequent iterations focused on refining ModelF by reducing multicollinearity while preserving essential predictive variables. The finalized set of variables in ModelF includes x1, x4, X9, x5, x6, XXiii, X7, x2, X8, x4X9, x1x4, x1x5, X7X9, x4x5, x6x6, x4X7, x1x2, x2X7, x6X7, x2x2, x1X9, x2x4, x5x5, x1X8, x4x4, x2XX, x1XXiii, x2X8, x1x6, x5x6, x6X9, x2x6, x2X9, x1X7, and X7XXiii.



```
Car.pred_ModelF  <-  Car.pred_train[,c('Y','x1','x4','X9','x5','x6','XXiii','X7','x2','X8','x4X9','x1x4','x1x5','X7X9','x4x5','x6x6','x4X7','x1x2','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x1X8','x4x4','x2XX','x:
Car.pred_ModelF_X  <-  Car.pred_train[c('x1','x4','X9','x5','x6','XXiii','X7','x2','X8','x4X9','x1x4','x1x5','X7X9','x4x5','x6x6','x4X7','x1x2','x2X7','x6X7','x2x2','x1X9','x2x4','x5x5','x1X8','x4x4','x2XX','x1XX:

head(Car.pred_ModelF)
colnames(Car.pred_ModelF)
nrow(Car.pred_ModelF)
p_F<-ncol(Car.pred_ModelF)
p_F
```

A data.frame: 6 × 36

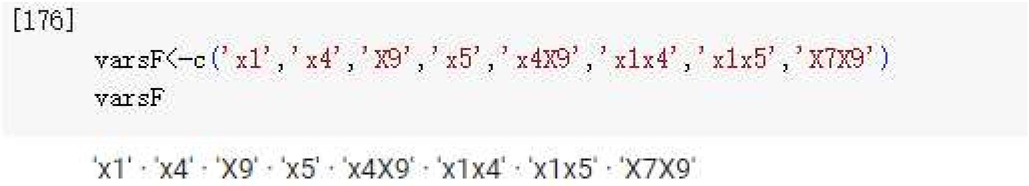
	Y	x1	x4	X9	x5	x6	XXiii	X7	x2	X8	x4X9	x1x4	x1x5	X7X9	x4x5	x6x6	x4X7	x1x2	x2X7
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-43.956253	0	-1.4282141	-55728.14	1	1	-4.1556228	0	0.000000	273.379748	8.8825772	0	62.7789408	3105625317	-43.956253	25.8453130	-4.1556228
1488	5.414973	0.7806403	-33.656253	0	-0.4282141	-40728.14	1	1	2.4843772	0	0.000000	-26.273428	-0.3342812	0	14.4120824	1658781190	-33.656253	1.9394050	2.4843772
2479	5.352183	-4.2193597	-2.056253	0	-0.4282141	-15728.14	1	0	0.1043772	1	0.000000	8.676072	1.8067893	0	0.8805166	247374311	0.000000	-0.4404049	0.0000000
1385	5.491362	2.7806403	-33.656253	0	-0.4282141	-30728.14	1	1	4.4443772	0	0.000000	-93.585934	-1.1907094	0	14.4120824	944218438	-33.656253	12.3582144	4.4443772
1992	5.342423	-4.2193597	-7.056253	1	-0.4282141	34271.86	1	1	0.8443772	0	-7.056253	29.772870	1.8067893	1	3.0215871	1174560555	-7.056253	-3.5627311	0.8443772
3827	6.021189	2.7806403	59.043747	1	1.5717859	-19728.14	0	0	-4.2556228	0	59.043747	164.179423	4.3705712	0	92.8041284	389199412	0.000000	-11.8333564	0.0000000

```
'Y' · 'x1' · 'x4' · 'X9' · 'x5' · 'x6' · 'XXiii' · 'X7' · 'x2' · 'X8' · 'x4X9' · 'x1x4' · 'x1x5' · 'X7X9' · 'x4x5' · 'x6x6' · 'x4X7' · 'x1x2' · 'x2X7' · 'x6X7' · 'x2x2' · 'x1X9' · 'x2x4' · 'x5x5' · 'x1X8' · 'x4x4' · 'x2XX' · 'x1XXiii' · 'x2X8' · 'x1x6' · 'x5x6' · 'x6X9' · 'x2x6' · 'x2X9' · 'x1X7' ·
'X7XXiii'
3198
36
```

ModelF Selection using Cp, BIC, R2 and Adj R2

This section focuses on the selection of the optimal ModelF using various criteria such as Cp, BIC, R2, and Adj R2 values.

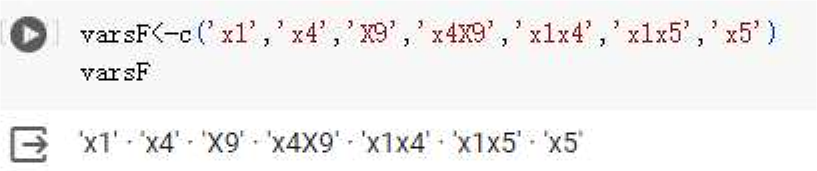
Analysis of Cp Values the Cp values were analyzed to determine the most suitable subset of predictors for ModelF. The analysis revealed that the top variables based on Cp criteria were x1, x4, X9, x5, x4X9, x1x4, x1x5, and X7X9. However, it was noted that interaction terms involving X7 should be removed due to the elimination of the root variable X7.



```
[176]
      varsF<-c('x1','x4','X9','x5','x4X9','x1x4','x1x5','X7X9')
      varsF

      'x1' · 'x4' · 'X9' · 'x5' · 'x4X9' · 'x1x4' · 'x1x5' · 'X7X9'
```

Analysis of BIC Values

Similarly, the BIC values were assessed to identify the optimal subset of predictors. The analysis indicated that the key variables according to BIC were x1, x4, X9, x4X9, x1x4, x1x5, and x5. This alignment with the Cp analysis further reinforced the importance of these predictors in ModelF.

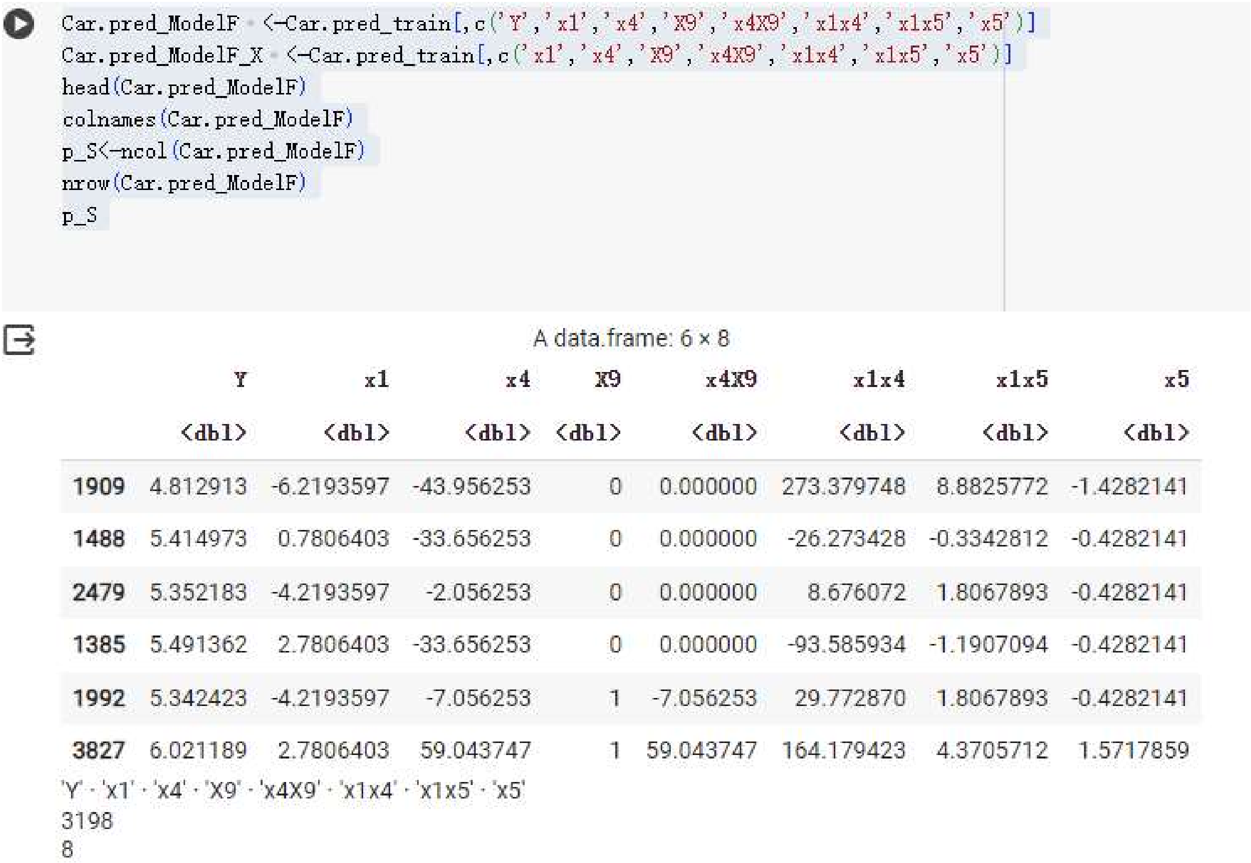


```
varsF<-c('x1','x4','X9','x4X9','x1x4','x1x5','x5')
varsF
```

```
'x1' · 'x4' · 'X9' · 'x4X9' · 'x1x4' · 'x1x5' · 'x5'
```

Analysis of R2 and Adj R2

The R2 and Adj R2 values were examined to gauge the explanatory power and goodness of fit of the model. The top variables based on these criteria were consistent with the Cp and BIC analyses, emphasizing the significance of x1, x4, X9, x4X9, x1x4, x1x5, and x5 in explaining the variation in the response variable Y.



```
Car.pred_ModelF <-Car.pred_train[,c('Y','x1','x4','X9','x4X9','x1x4','x1x5','x5')]
Car.pred_ModelF_X <-Car.pred_train[,c('x1','x4','X9','x4X9','x1x4','x1x5','x5')]
head(Car.pred_ModelF)
colnames(Car.pred_ModelF)
p_S<-ncol(Car.pred_ModelF)
nrow(Car.pred_ModelF)
p_S
```

A data.frame: 6 × 8

	Y	x1	x4	X9	x4X9	x1x4	x1x5	x5
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1909	4.812913	-6.2193597	-43.956253	0	0.000000	273.379748	8.8825772	-1.4282141
1488	5.414973	0.7806403	-33.656253	0	0.000000	-26.273428	-0.3342812	-0.4282141
2479	5.352183	-4.2193597	-2.056253	0	0.000000	8.676072	1.8067893	-0.4282141
1385	5.491362	2.7806403	-33.656253	0	0.000000	-93.585934	-1.1907094	-0.4282141
1992	5.342423	-4.2193597	-7.056253	1	-7.056253	29.772870	1.8067893	-0.4282141
3827	6.021189	2.7806403	59.043747	1	59.043747	164.179423	4.3705712	1.5717859

```
'Y' · 'x1' · 'x4' · 'X9' · 'x4X9' · 'x1x4' · 'x1x5' · 'x5'
3198
8
```

ModelB Selection using Cp, BIC, R2 and Adj R2

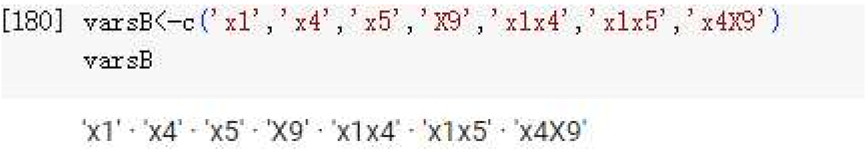
This section focuses on the selection of the optimal ModelB using Cp, BIC, R2, and Adj R2 criteria.

Analysis and Selection of Variables

The process involved analyzing the Cp, BIC, R2, and Adj R2 values for ModelB to determine the most influential variables. The analysis suggested the inclusion of 'x1', 'x4', 'x5', 'X9', 'x1x4', 'x1x5', and 'x4X9' as significant predictors for ModelB. However, 'x2X7' was excluded due to missing root variables.

Conclusion

Upon comparison with ModelF, it was noted that ModelB shares the same variables. Therefore, further analysis or modifications were deemed unnecessary for ModelB.



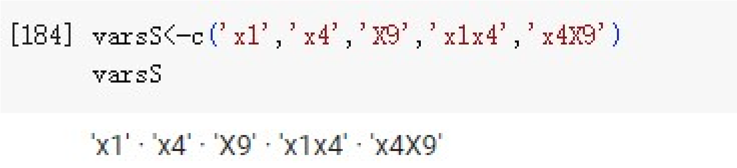
```
[180] varsB<-c('x1','x4','x5','X9','x1x4','x1x5','x4X9')
      varsB

      'x1' · 'x4' · 'x5' · 'X9' · 'x1x4' · 'x1x5' · 'x4X9'
```

ModelS Selection using Cp, BIC, R2 and Adj R2

This section focuses on the selection of the optimal ModelS using Cp, BIC, R2, and Adj R2 criteria.

Analysis and Selection of Variables

The analysis involved evaluating Cp, BIC, R2, and Adj R2 values to determine the most relevant variables for ModelS. The best model, which includes 'x1', 'x4', 'X9', 'x4X9', and 'x1x4', was identified. These variables were deemed significant for predicting the response variable Y within ModelS.



```
[184] varsS<-c('x1','x4','X9','x1x4','x4X9')
      varsS
```

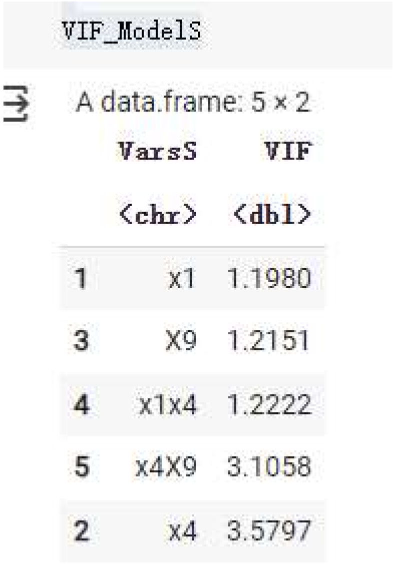
```
'x1' · 'x4' · 'X9' · 'x1x4' · 'x4X9'
```

Multicollinearity Test - VIF- ModelS

This section focuses on assessing multicollinearity using the Variance Inflation Factor (VIF) for ModelS.

ModelS - VIF Test - Iteration 1

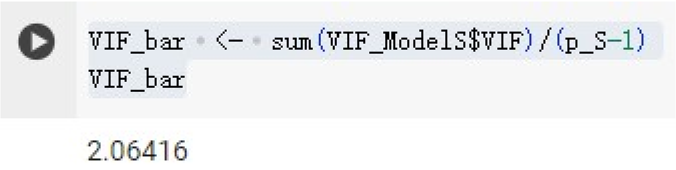
The VIF test was conducted for ModelS, which includes the predictors 'x1', 'x4', 'X9', 'x1x4', and 'x4X9'. The VIF values were computed to evaluate the level of multicollinearity between these variables.



```
VIF_ModelS
```

A data.frame: 5 × 2

	VarsS	VIF
	<chr>	<dbl>
1	x1	1.1980
3	X9	1.2151
4	x1x4	1.2222
5	x4X9	3.1058
2	x4	3.5797



```
VIF_bar <- sum(VIF_ModelS$VIF)/(p_S-1)
VIF_bar
```

```
2.06416
```

Conclusion

The analysis revealed low VIF values, particularly highlighted by the low VIF_bar and individual VIF values. This indicates minimal multicollinearity among the variables included in ModelS. The absence of significant multicollinearity supports the model's robustness and the independence of the predictors in explaining the response variable Y.

Multiple Linear Regression

This section focuses on fitting variables into MLR models and analyzing their correlations.

ModelF

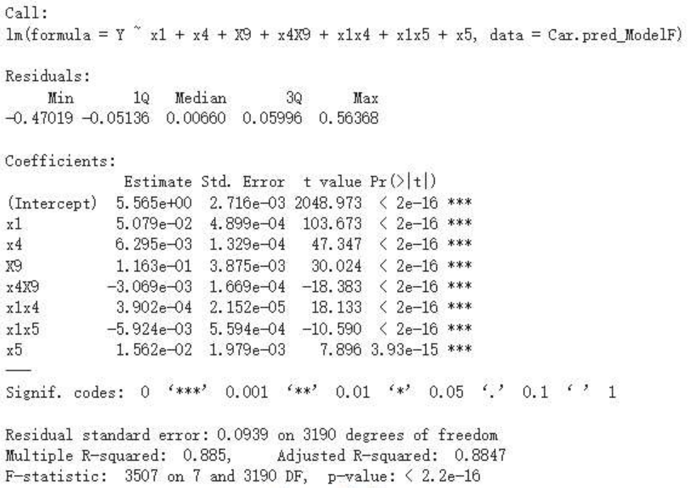
Fitting the variables obtained into a multiple linear regression model, the analysis reveals:

The model includes 'x1', 'x4', 'X9', 'x4X9', 'x1x4', 'x1x5', and 'x5'.

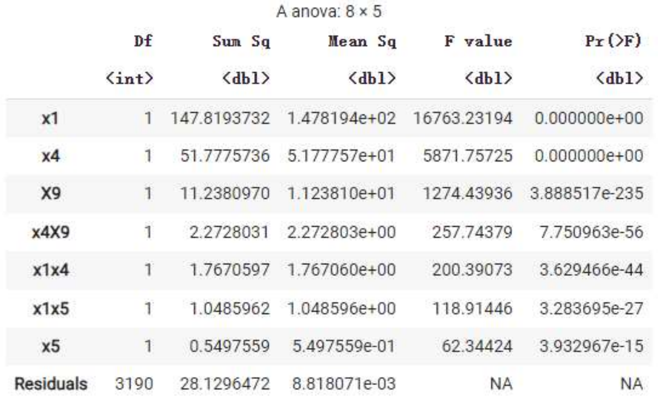
The model summary shows all eight variables with a P-value close to zero, indicating their significance in predicting the response variable Y.

The R-squared value and adjusted R-squared values around 0.8 indicate a good fit for the model.

The correlation matrix highlights 'x1' as the highly correlated variable with Y, while 'x4' and 'x4X9' exhibit a correlation of 0.8.

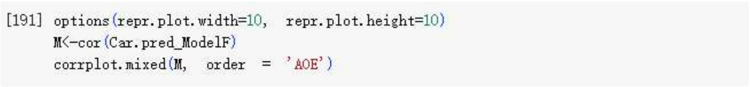


```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x4X9 + x1x4 + x1x5 + x5, data = Car.pred_ModelF)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.47019 -0.05136  0.00660  0.05996  0.56368 

Coefficients:
              Estimate Std. Error  t value Pr(>|t|)    
(Intercept)  5.565e+00  2.716e-03 2048.973  < 2e-16 ***
x1           5.079e-02  4.899e-04  103.673  < 2e-16 ***
x4           6.295e-03  1.329e-04   47.347  < 2e-16 ***
X9           1.163e-01  3.875e-03   30.024  < 2e-16 ***
x4X9        -3.069e-03  1.669e-04  -18.383  < 2e-16 ***
x1x4         3.902e-04  2.152e-05   18.133  < 2e-16 ***
x1x5        -5.924e-03  5.594e-04  -10.590  < 2e-16 ***
x5           1.562e-02  1.979e-03    7.896 3.93e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0939 on 3190 degrees of freedom
Multiple R-squared:  0.885,	Adjusted R-squared:  0.8847 
F-statistic:  3507 on 7 and 3190 DF,  p-value: < 2.2e-16
```

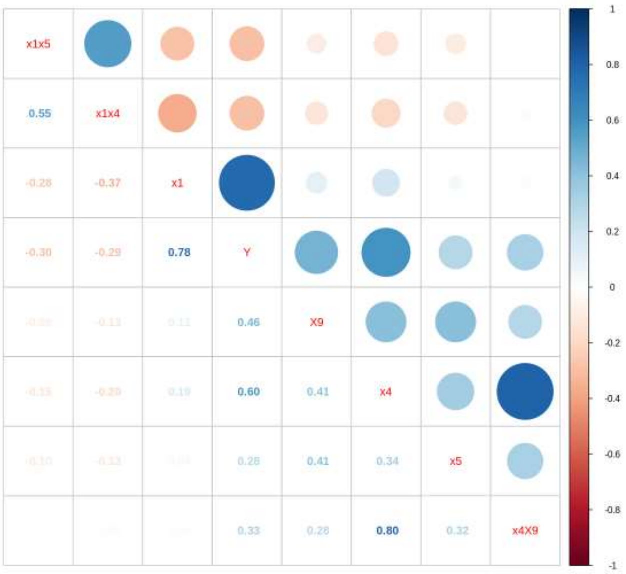


A anova: 8 × 5

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
	<int>	<dbl>	<dbl>	<dbl>	<dbl>
x1	1	147.8193732	1.478194e+02	16763.23194	0.000000e+00
x4	1	51.7775736	5.177757e+01	5871.75725	0.000000e+00
X9	1	11.2380970	1.123810e+01	1274.43936	3.888517e-235
x4X9	1	2.2728031	2.272803e+00	257.74379	7.750963e-56
x1x4	1	1.7670597	1.767060e+00	200.39073	3.629466e-44
x1x5	1	1.0485962	1.048596e+00	118.91446	3.283695e-27
x5	1	0.5497559	5.497559e-01	62.34424	3.932967e-15
Residuals	3190	28.1296472	8.818071e-03	NA	NA



```
[191] options(repr.plot.width=10,  repr.plot.height=10)
      M<-cor(Car.pred_ModelF)
      corrplot.mixed(M,  order  =  'AOE')
```



ModelS

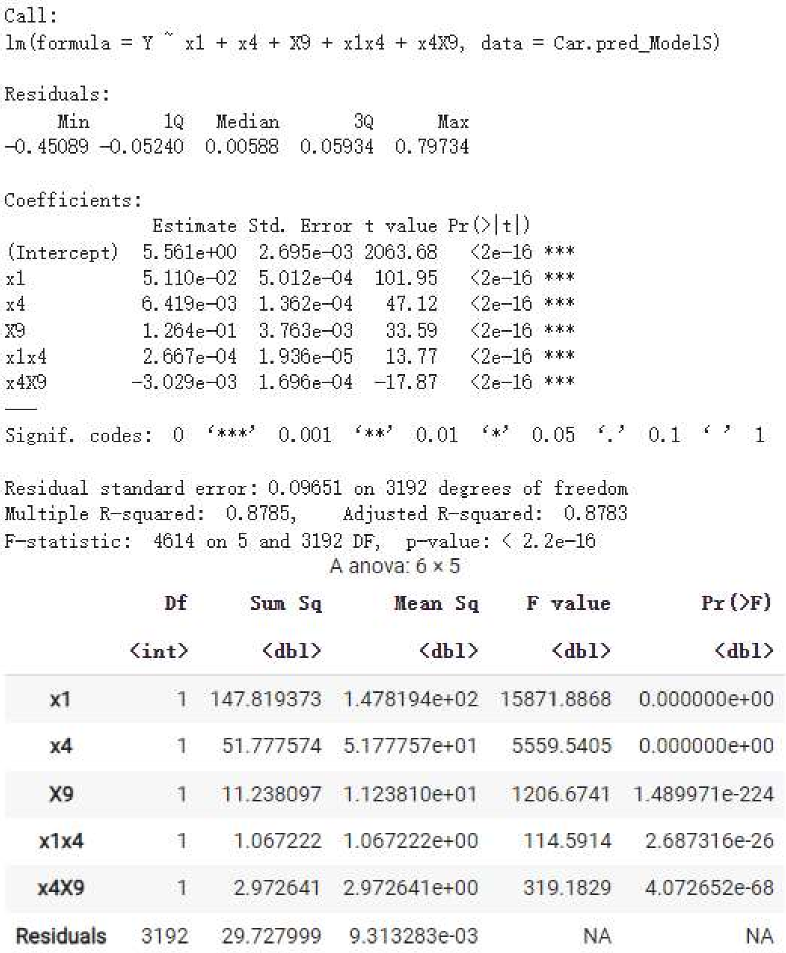
Similarly, for ModelS, the analysis reveals:

The model includes 'x1', 'x4', 'X9', 'x4X9', and 'x1x4'.

The model summary shows all five variables with a P-value close to zero, indicating their significance in predicting Y.

The R-squared value and adjusted R-squared values around 0.878 indicate a very good fit for the model.

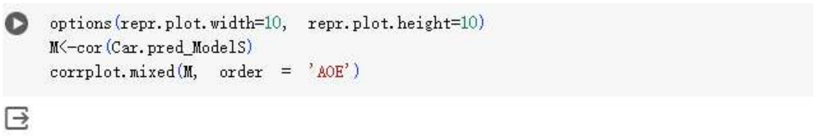
The correlation matrix shows a high correlation of 0.8 between 'x4' and 'x4X9'.



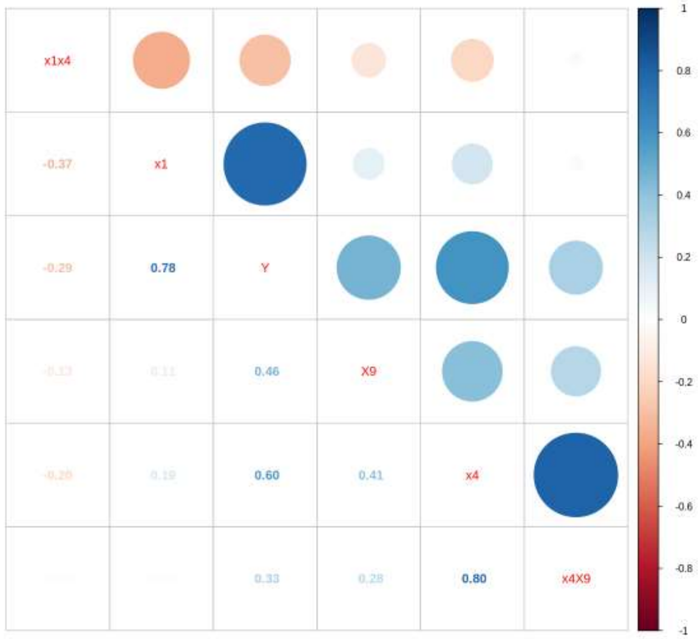
```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x1x4 + x4X9, data = Car.pred_ModelS)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.45089 -0.05240  0.00588  0.05934  0.79734 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  5.561e+00  2.695e-03 2063.68   <2e-16 ***
x1           5.110e-02  5.012e-04  101.95   <2e-16 ***
x4           6.419e-03  1.362e-04   47.12   <2e-16 ***
X9           1.264e-01  3.763e-03   33.59   <2e-16 ***
x1x4         2.667e-04  1.936e-05   13.77   <2e-16 ***
x4X9        -3.029e-03  1.696e-04  -17.87   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09651 on 3192 degrees of freedom
Multiple R-squared:  0.8785,	Adjusted R-squared:  0.8783 
F-statistic:  4614 on 5 and 3192 DF,  p-value: < 2.2e-16
```

A anova: 6 × 5

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
	<int>	<dbl>	<dbl>	<dbl>	<dbl>
x1	1	147.819373	1.478194e+02	15871.8868	0.000000e+00
x4	1	51.777574	5.177757e+01	5559.5405	0.000000e+00
X9	1	11.238097	1.123810e+01	1206.6741	1.489971e-224
x1x4	1	1.067222	1.067222e+00	114.5914	2.687316e-26
x4X9	1	2.972641	2.972641e+00	319.1829	4.072652e-68
Residuals	3192	29.727999	9.313283e-03	NA	NA



```
options(repr.plot.width=10,  repr.plot.height=10)
M<-cor(Car.pred_ModelS)
corrplot.mixed(M,  order  =  'AOE')
```



Extra Sum of Squares

In this section, we analyze the significance of the variable 'x4X9' using the Extra Sum of Squares method.

Analysis Steps:

Variables selected: 'x4X9,' which exhibits a high correlation with 'x4.'

Total Sum of Squares (SST):

SST is calculated as the sum of squared differences between each observation and the mean of the response variable Y.

Full Model (ModelF):

The full model includes all variables: 'x1,' 'x4,' 'X9,' 'x4X9,' 'x1x4,' 'x1x5,' and 'x5.'

An analysis of variance (ANOVA) is performed for the full model.

Model with Other Variables:

Another model is created without the variable 'x4X9' to compare its impact.

ANOVA is also conducted for this model.

Extra Sums of Squares Analysis:

The extra sum of squares (SSRx4X9_OtherVars) is computed to assess the contribution of 'x4X9' beyond the other variables.

The F* statistic is calculated to test the hypothesis regarding the coefficient of 'x4X9.'

Hypothesis and Conclusion

Hypothesis:

Null Hypothesis (H0): The regression coefficient of 'x4X9' is zero.

Alternative Hypothesis (Ha): The regression coefficient of 'x4X9' is not zero.

Analysis Plan:

Significance Level: α = 0.10

Test Method: F Test F*

Sample Data Analysis:

F* value is compared with the critical F-value to make the decision.

Conclusion

Based on the analysis:

The F* value is greater than the critical F-value.

Therefore, we reject the null hypothesis (H0) and conclude that the variable 'x4X9' is significant in the regression model.

ModelF/ ModelS Outlier and Regression Analysis

Outliers Testing:

Identify potential outliers that can significantly impact the regression model's results and interpretation. Used the outlierTest function for outlier testing and to understand the potential impact of outliers on the regression model.

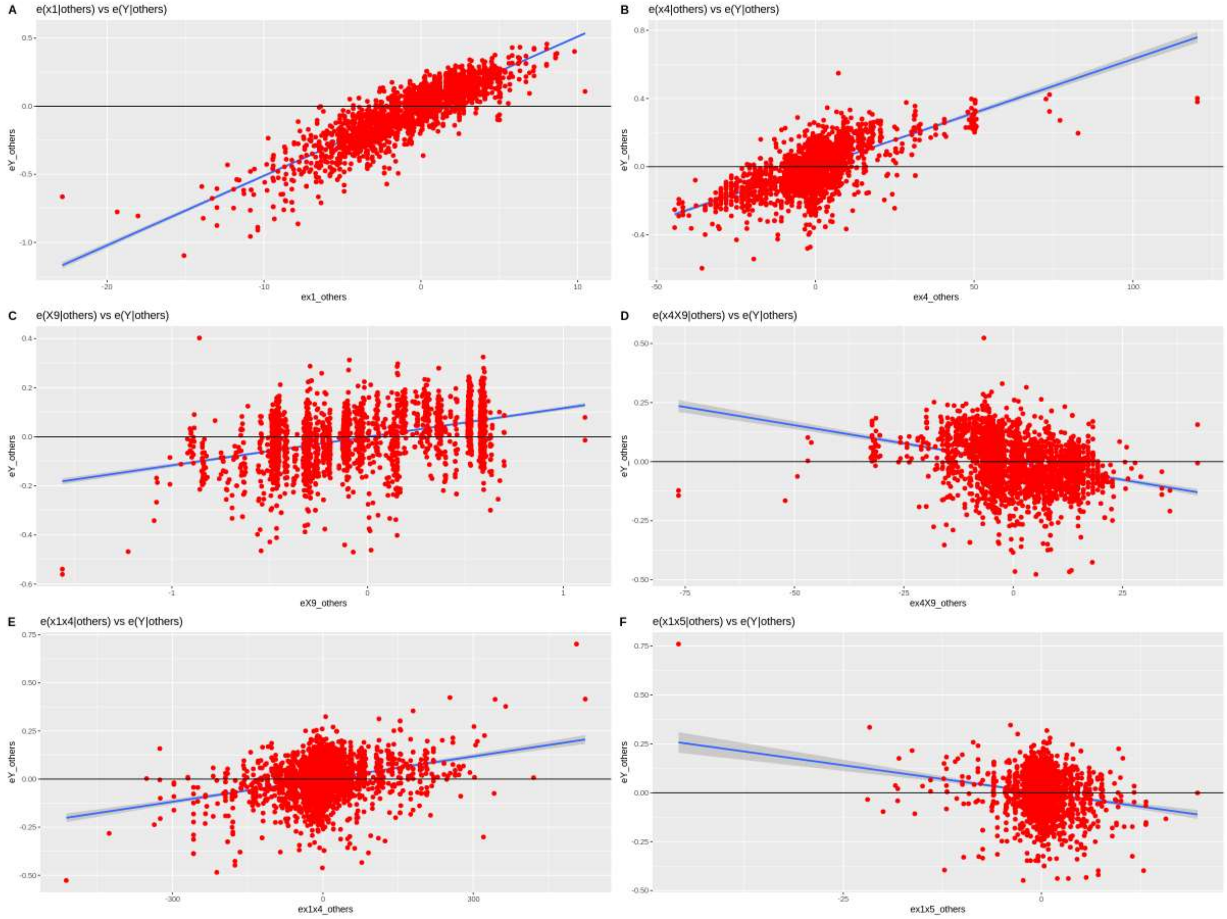
Influential Cases:

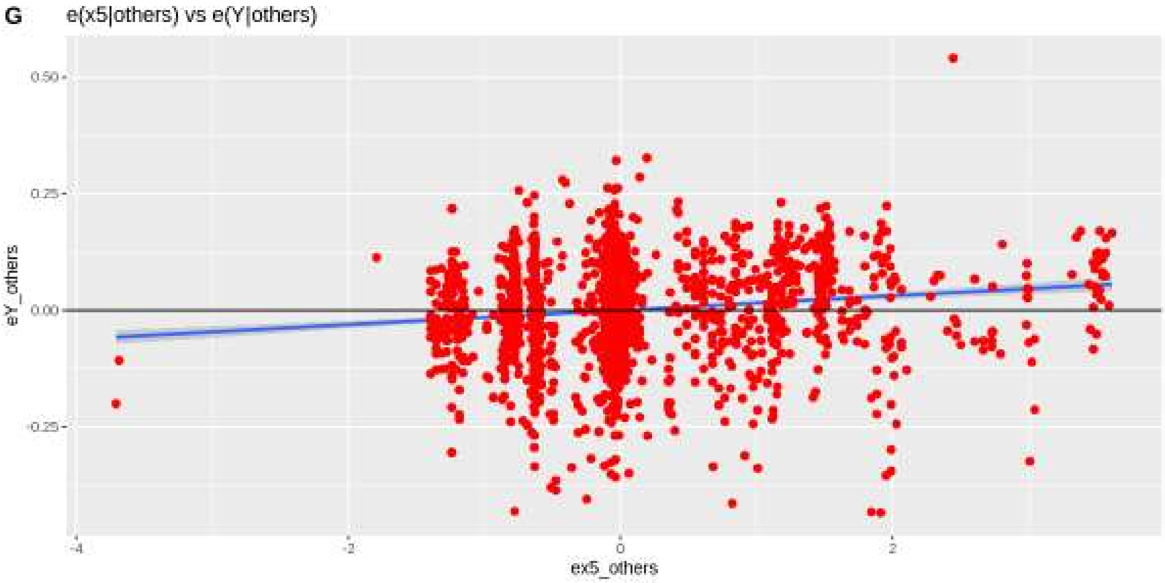
Assess the influence of individual data points on the regression model's parameters and predictions. Employed the influencePlot function to visualize influential cases, helping identify data points that significantly affect the regression model coefficients.

Added Variable Plots:

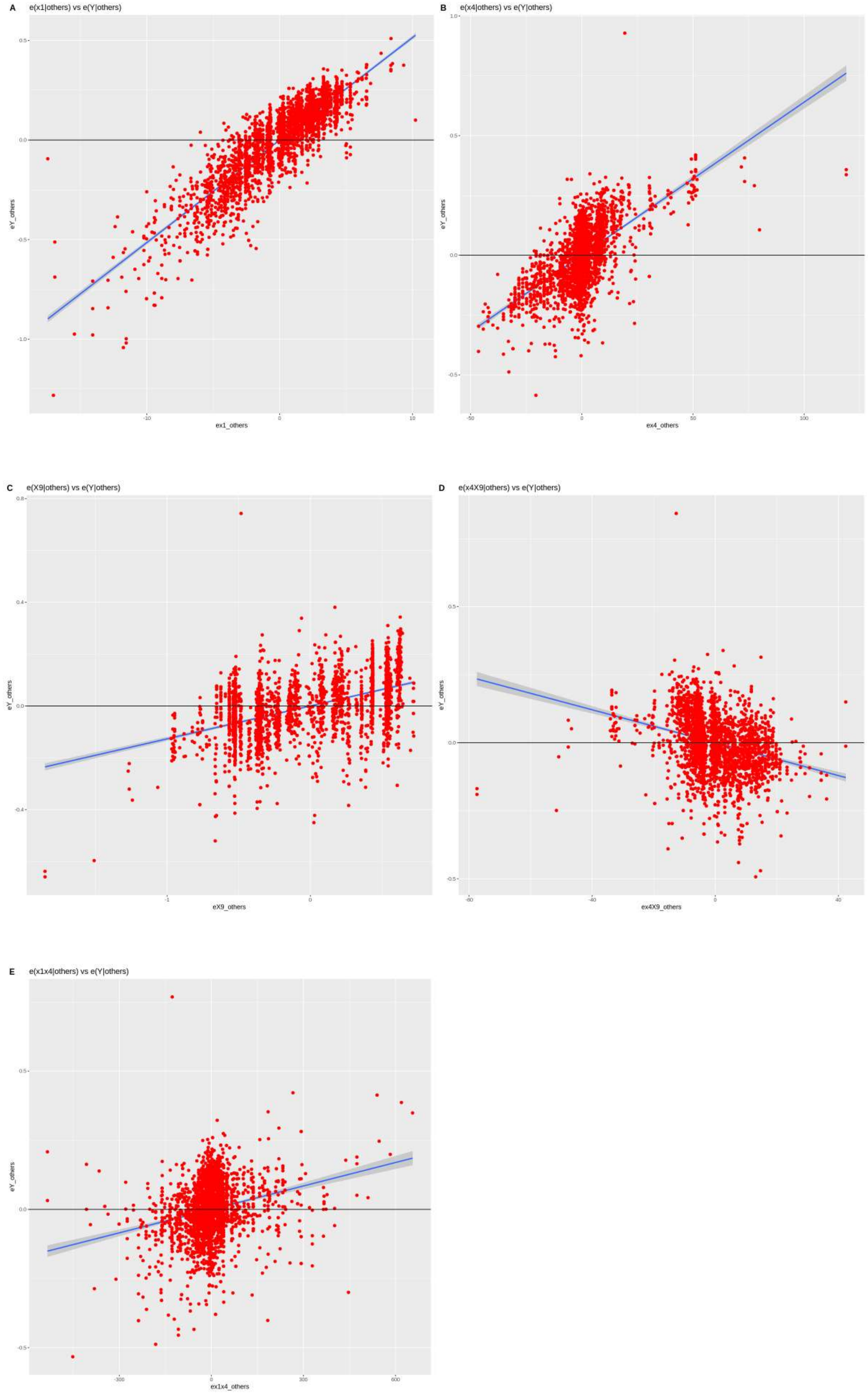
Examine the relationship between the response variable and each predictor variable while controlling for other predictors. Utilized variable plots created using the Added Variable Plots function to examine the relationships between each predictor variable and the response variable, aiding in detecting nonlinearity or influential observations.

Added Variable Plots of ModelF:





Added Variable Plots of ModelS:


A
e(x1|others) vs e(Y|others)
eY_others
0.5
0.0
-0.5
-1.0
-10
0
10
ex1_others
B
e(x4|others) vs e(Y|others)
eY_others
1.0
0.5
0.0
-0.5
-50
0
50
100
ex4_others
C
e(X9|others) vs e(Y|others)
eY_others
0.8
0.4
0.0
-0.4
-1
0
eX9_others
D
e(x4X9|others) vs e(Y|others)
eY_others
0.5
0.0
-0.5
-80
-40
0
40
ex4X9_others
E
e(x1x4|others) vs e(Y|others)
eY_others
0.5
0.0
-0.5
-300
0
300
600
ex1x4_others

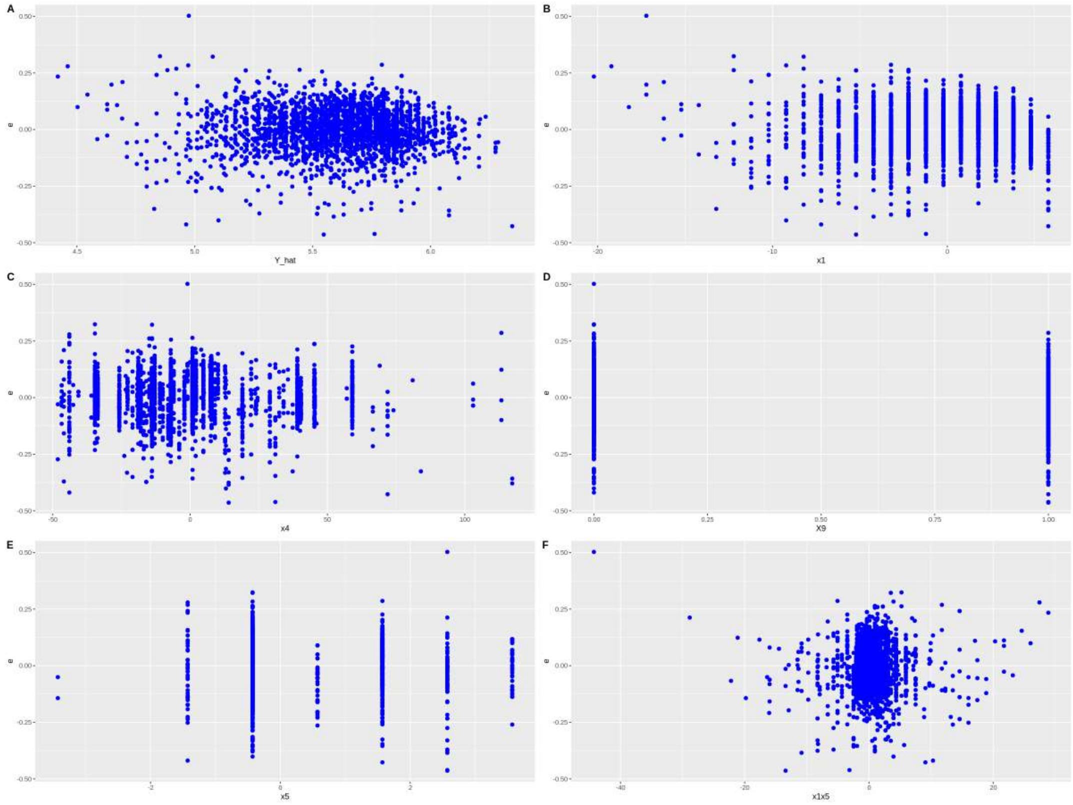
Residual Analysis:

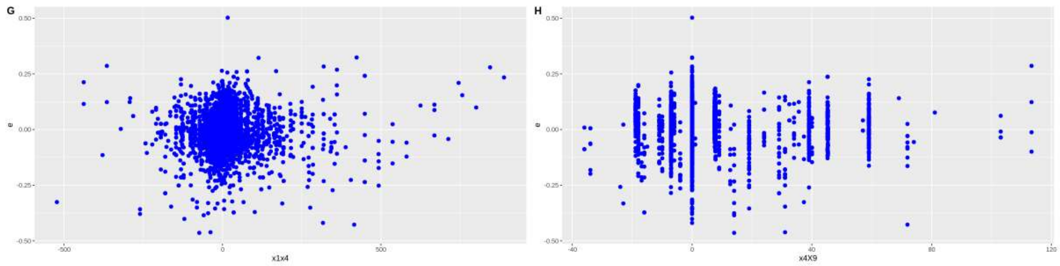
Evaluate the differences between observed and predicted values to ensure the model's assumptions are met and the residuals are unbiased and normally distributed. Conducted residual analysis using the residualPlots function to evaluate patterns such as heteroscedasticity or nonlinearity in residuals, which can affect the model's accuracy.

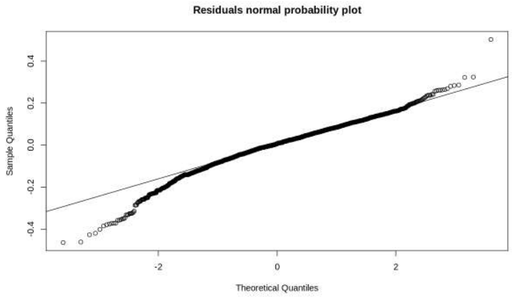
Diagnostic Tests:

Conduct various tests to check for violations of assumptions and assess the overall goodness of fit of the regression model. Performed various diagnostic tests, including the Breusch-Pagan test for heteroscedasticity and the RESET test for functional form misspecification, to assess overall model fit and reliability.

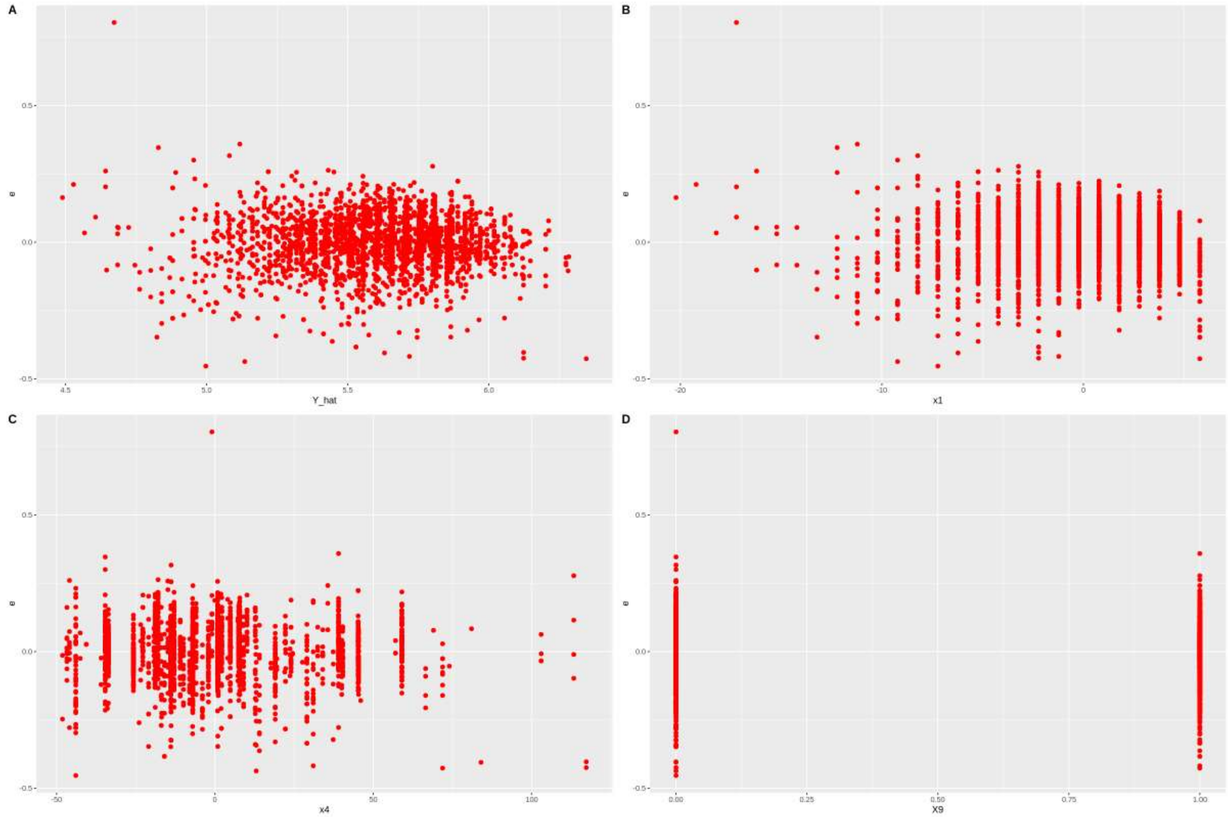
Diagnostic Plots of ModelF:

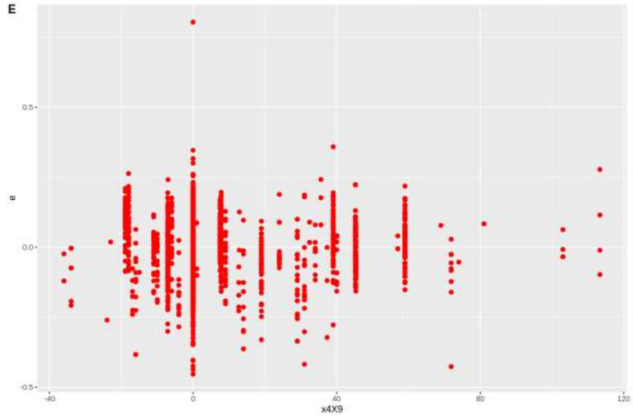


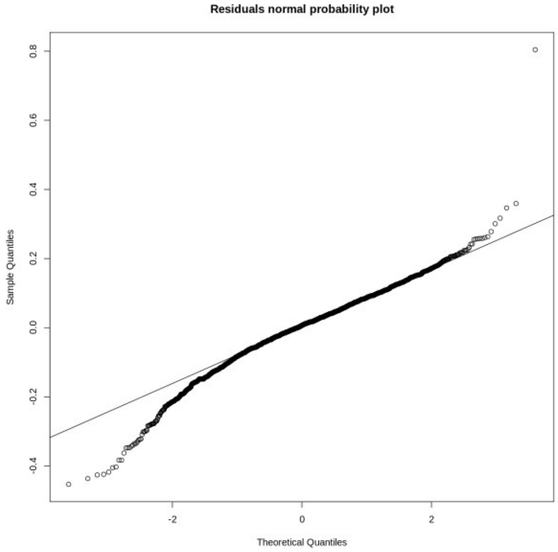




Diagnostic Plots of ModelS:







Normality Testing:

Assess the normality of residuals to ensure that the errors follow a normal distribution, which is crucial for the reliability of the regression model. Checked the normality assumption of residuals using normality tests like the Shapiro-Wilk test or visual inspection through QQ plots, which is crucial for the validity of regression analysis.

Final MLR ModelF

In this section, we present the finalized Multiple Linear Regression (MLR) ModelF based on the selected variables.

The regression function derived from the finalized MLR ModelF is as follows:

$Y = 5.565 + 0.051 \ast x1 + 0.006 \ast x4 + 0.116 \ast X9 + (-0.003) \ast x4X9 + 0.00039 \ast x1x4 + (-0.0056) \ast x1x5 + 0.0154 \ast x5$

Model Performance

The performance metrics for the finalized MLR ModelF are as follows:

Residual standard error: 0.09327

Adjusted R-squared: 0.8847

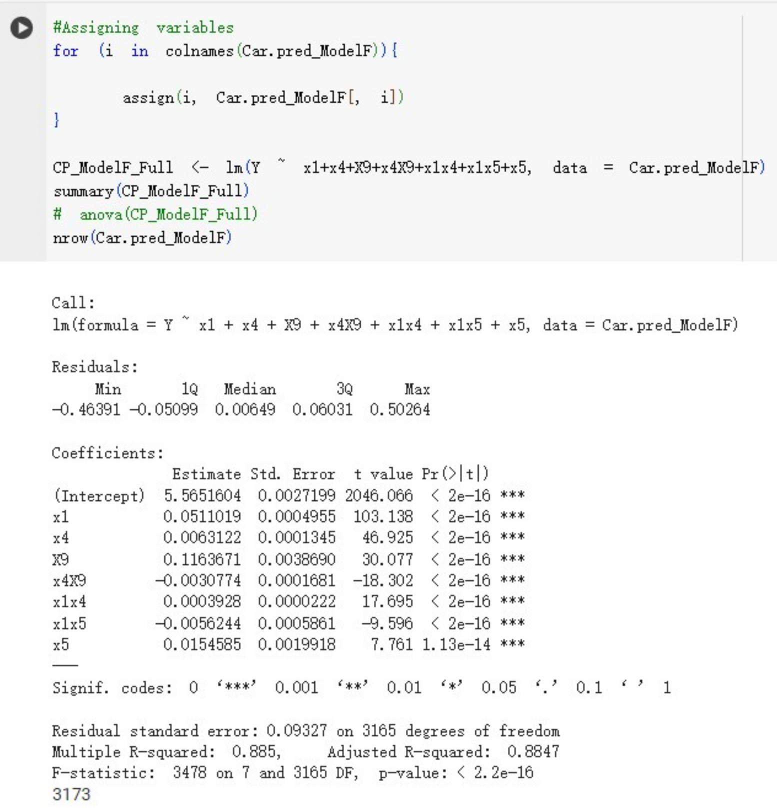
These metrics indicate that the model explains approximately 88.47% of the variance in the dependent variable.

Dataset Details

Number of Rows in the training set: 3173

Number of variables in the function: 07

These details provide context regarding the size of the dataset and the complexity of the regression model.



```
#Assigning  variables
for  (i  in  colnames(Car.pred_ModelF)){

       assign(i,  Car.pred_ModelF[,  i])
}

CP_ModelF_Full  <-  lm(Y  ~  x1+x4+X9+x4X9+x1x4+x1x5+x5,  data  =  Car.pred_ModelF)
summary(CP_ModelF_Full)
#  anova(CP_ModelF_Full)
nrow(Car.pred_ModelF)
```

```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x4X9 + x1x4 + x1x5 + x5, data = Car.pred_ModelF)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.46391 -0.05099  0.00649  0.06031  0.50264 

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)    
(Intercept)  5.5651604  0.0027199 2046.066  < 2e-16 ***
x1           0.0511019  0.0004955  103.138  < 2e-16 ***
x4           0.0063122  0.0001345   46.925  < 2e-16 ***
X9           0.1163671  0.0038690   30.077  < 2e-16 ***
x4X9        -0.0030774  0.0001681  -18.302  < 2e-16 ***
x1x4         0.0003928  0.0000222   17.695  < 2e-16 ***
x1x5        -0.0056244  0.0005861   -9.596  < 2e-16 ***
x5           0.0154585  0.0019918    7.761 1.13e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09327 on 3165 degrees of freedom
Multiple R-squared:  0.885,	Adjusted R-squared:  0.8847 
F-statistic:  3478 on 7 and 3165 DF,  p-value: < 2.2e-16
3173
```

Model validation of ModelF

In this section, we perform model validation for ModelF using a separate test sample dataset.

Regression Function Validation

The regression function obtained when fitted using the test sample dataset is as follows:

$Y = 5.564 + 0.050 \ast x1 + 0.00627 \ast x4 + 0.1210 \ast X9 + -0.00312 \ast x4X9 + 0.00038 \ast x1x4 + -0.00256 \ast x1x5 + 0.02101 \ast x5$

Model Performance on Validation

The performance metrics for the validated ModelF are as follows:

Residual standard error: 0.0961

Adjusted R-squared: 0.8824

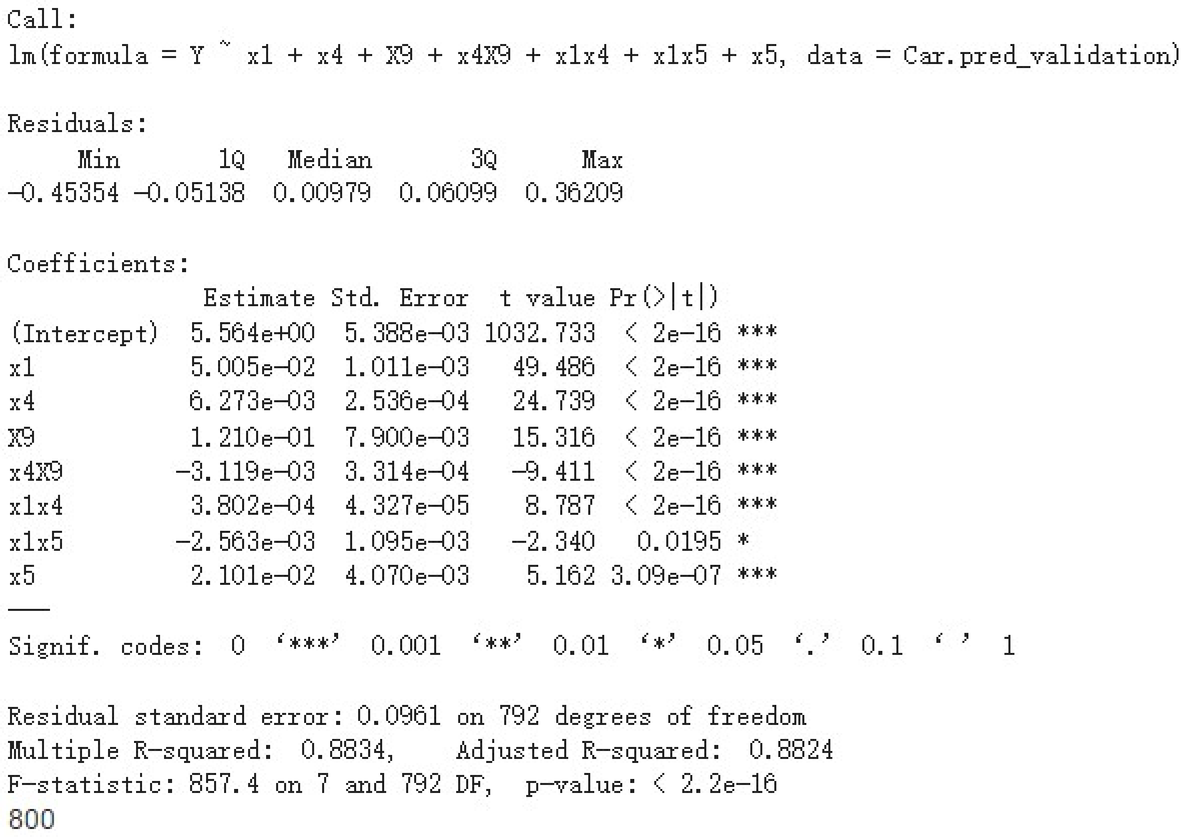
These metrics indicate that the model explains approximately 88.24% of the variance in the dependent variable in the validation dataset.

Dataset Details

Number of Rows in the test set: 800

Number of variables in the function: 07

The comparable values of the validation model to the train model suggest that the ModelF is robust and performs well on unseen data.



```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x4X9 + x1x4 + x1x5 + x5, data = Car.pred_validation)

Residuals:
     Min       1Q   Median       3Q      Max
-0.45354 -0.05138  0.00979  0.06099  0.36209

Coefficients:
              Estimate Std. Error  t value Pr(>|t|)
(Intercept)  5.564e+00  5.388e-03 1032.733  < 2e-16 ***
x1           5.005e-02  1.011e-03   49.486  < 2e-16 ***
x4           6.273e-03  2.536e-04   24.739  < 2e-16 ***
X9           1.210e-01  7.900e-03   15.316  < 2e-16 ***
x4X9        -3.119e-03  3.314e-04   -9.411  < 2e-16 ***
x1x4         3.802e-04  4.327e-05    8.787  < 2e-16 ***
x1x5        -2.563e-03  1.095e-03   -2.340   0.0195 *
x5           2.101e-02  4.070e-03    5.162 3.09e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.0961 on 792 degrees of freedom
Multiple R-squared:  0.8834,    Adjusted R-squared:  0.8824
F-statistic: 857.4 on 7 and 792 DF,  p-value: < 2.2e-16
800
```

Final MLR ModelS

In this section, we present the final multiple linear regression model for ModelS.

Regression Function

The regression function obtained for ModelS is as follows:

Y = 5.561 + 0.05148 ∗ x1 + 0.006398 ∗ x4+ 0.1269 ∗ X9+ -0.003024 ∗ x4X9+ 0.0002826 ∗ x1x4

Model Performance

Residual standard error: 0.09512

Adjusted R-squared: 0.8795

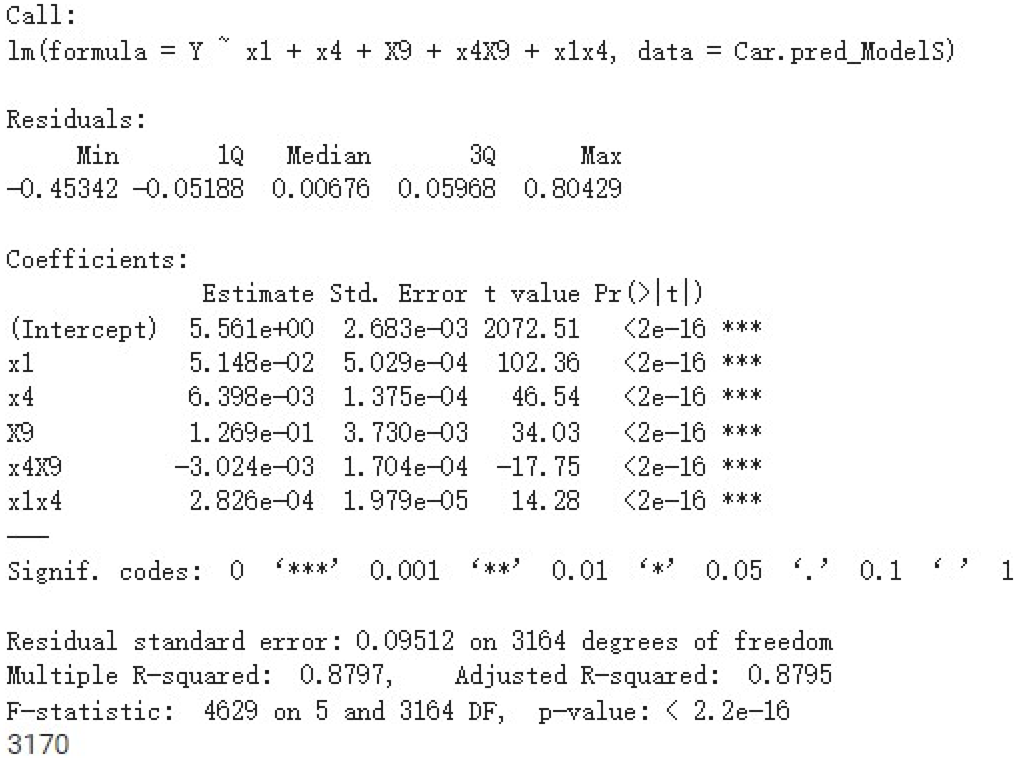
The adjusted R-squared value of 0.8795 indicates that approximately 87.95% of the variance in the dependent variable is explained by the independent variables in the model.

Dataset Details

Number of Rows in the train set: 3170

Number of variables in the function: 05

The model includes five variables and demonstrates good performance in capturing the relationship between the predictors and the target variable.



```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x4X9 + x1x4, data = Car.pred_ModelS)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.45342 -0.05188  0.00676  0.05968  0.80429 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  5.561e+00  2.683e-03 2072.51   <2e-16 ***
x1           5.148e-02  5.029e-04  102.36   <2e-16 ***
x4           6.398e-03  1.375e-04   46.54   <2e-16 ***
X9           1.269e-01  3.730e-03   34.03   <2e-16 ***
x4X9        -3.024e-03  1.704e-04  -17.75   <2e-16 ***
x1x4         2.826e-04  1.979e-05   14.28   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.09512 on 3164 degrees of freedom
Multiple R-squared:  0.8797,	Adjusted R-squared:  0.8795 
F-statistic:  4629 on 5 and 3164 DF,  p-value: < 2.2e-16
3170
```

Model Validation of ModelS

In this section, we validate the ModelS using the test sample dataset.

Regression Function for ModelS Validation

The regression function obtained when fitting the ModelS using the test sample dataset is:

$$Y = 5.558 + 0.049 \ast x1 + 0.00633 \ast x4 + 0.133 \ast X9 + -0.00294 \ast x4X9 + 0.00029 \ast x1x4$$

Model Performance Metrics

Residual standard error: 0.09829

Adjusted R-squared: 0.877

The adjusted R-squared value of 0.877 indicates that approximately 87.7% of the variance in the dependent variable is explained by the independent variables in the ModelS.

Dataset Details

Number of Rows in the test set: 800

Number of variables in the function: 05

Comparison with ModelF

The Adjusted R-squared values for ModelF and ModelS are as follows:

Adjusted R-squared of ModelF: 0.882

Adjusted R-squared of ModelS: 0.877

Since both models have comparable R-squared values, the 5-variable model (ModelS) is preferred over the 7-variable model (ModelF).

Coefficients in Original Scale

After returning the ModelS coefficients back to their original scale, the regression function becomes:

Y = -52.567 + 0.0743 ∗ X1 + 0.5756 ∗ X4+ 0.3717 ∗ X9+ -0.003024 ∗ X4X9+ 0.0002826 ∗ X1X4

The most significant variables and interaction terms included in the regression function are:

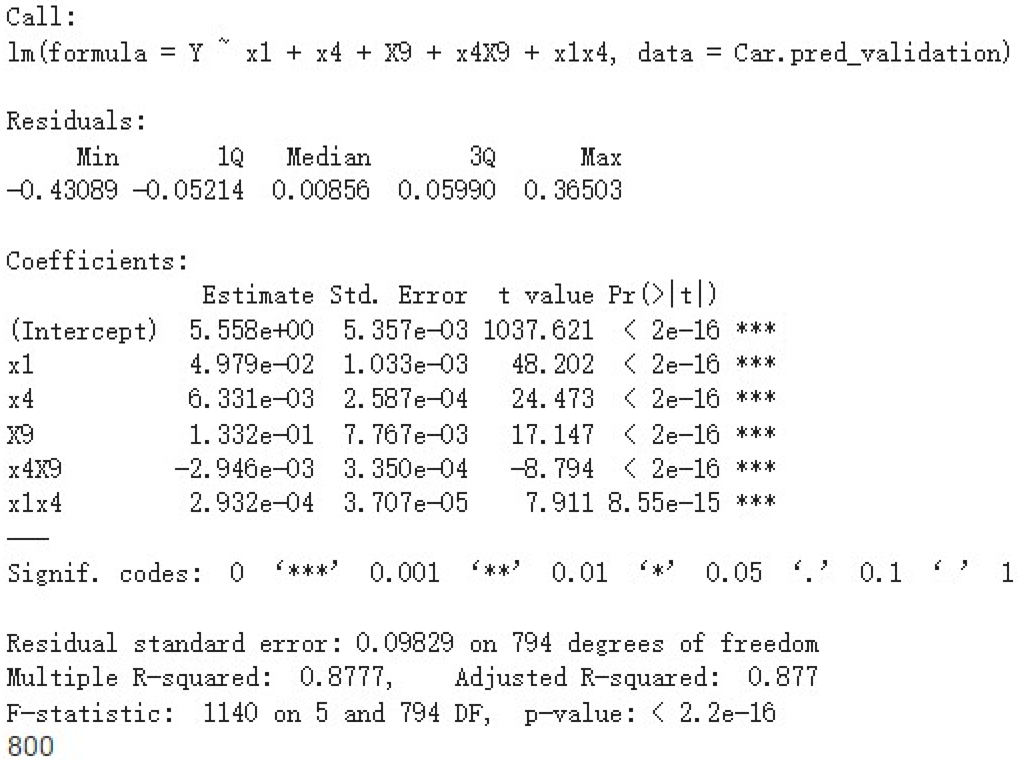
1. X1 Year

2. X4 Max_Power

3. X9 Fuel:Deisel

4. X4X9 Max_Power, Fuel:Deisel

5. X1X4 Year, Max_Power



```
Call:
lm(formula = Y ~ x1 + x4 + X9 + x4X9 + x1x4, data = Car.pred_validation)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.43089 -0.05214  0.00856  0.05990  0.36503 

Coefficients:
              Estimate Std. Error  t value Pr(>|t|)    
(Intercept)  5.558e+00  5.357e-03 1037.621  < 2e-16 ***
x1           4.979e-02  1.033e-03   48.202  < 2e-16 ***
x4           6.331e-03  2.587e-04   24.473  < 2e-16 ***
X9           1.332e-01  7.767e-03   17.147  < 2e-16 ***
x4X9        -2.946e-03  3.350e-04   -8.794  < 2e-16 ***
x1x4         2.932e-04  3.707e-05    7.911 8.55e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09829 on 794 degrees of freedom
Multiple R-squared:  0.8777,    Adjusted R-squared:  0.877 
F-statistic:  1140 on 5 and 794 DF,  p-value: < 2.2e-16
800
```

Practical Significance of Models and Variables

For someone looking to buy a preowned car, what do the significant variables in the data model mean in real time? This means that car buyers can more accurately assess the actual value and performance of a used car. The significance of our model lies in its ability to guide users in purchasing preowned cars or assist preowned car dealers in estimating car prices. Based on the variables we've retained, the Year variable reflects the vehicle's usage time and technological level, thereby affecting the vehicle's depreciation rate and reliability. Max Power directly relates to the car's power performance and driving experience, crucial for buyers seeking handling and power. Diesel fuel type often indicates better fuel economy and torque performance, meaning buyers can save fuel costs and enjoy a better driving experience in daily use. Regarding cross variables, such as the combination of Max Power and fuel type, as well as Year and Max Power, they provide more comprehensive and practical information, helping buyers fully understand the vehicle's performance and adaptability. Therefore, these significant variables are not just numbers; they directly reflect the real-time needs and decisions of car buyers, enabling them to make wiser choices when purchasing a used car. This guidance is crucial for buyers in selecting a vehicle that suits their needs and for sellers to price their vehicles reasonably.

References:

Dataset link: https://www.kaggle.com/datasets/sukhmandeepsinghbrar/car-price-prediction-dataset

R Code link in Google Colab:

https://colab.research.google.com/drive/1ez1XrL2CaOIw7aFiJrwTo4L3dJSW_WkM?usp=sharing