Using Multiple Linear Regression to Predict Car Prices

Nathanael Nam

5 Predictors Used

14 Betas, including B0

BIC Score: -21854.05

Abstract:
The goal of this research is to create a regression model to predict the price of a new car for the Chinese automobile company "Geely Auto," which is attempting to expand its manufacturing to U.S. markets to compete with U.S. and European markets. By understanding which factors and independent variables have an effect on the car price in a U.S. market, our model will allow the company to price their cars accurately according to our regression model. In this regression model, we want to get the highest predictive power while also having the least number of predictor coefficients in the regression model to have a somewhat simple model that can be easily understood. We will be using R studio as our program to create, test, and evaluate the model.
We are given two datasets of cars, training, and testing. They have the same variables except for PriceNew, which we are supposed to predict using the model we create from the training data. We will create our multiple linear regression model by testing various combinations of numerical and categorical predictors. We will also make sure to check conditions including heteroscedasticity, multicollinearity, linearity, normality of errors, and independence of errors. Our final model will have Horsepower + Length + MakeNew + AirBags + Type as our predictors that are most important in determining the price of the car. In the Kaggle competition, my name is listed as Nathanael Nam. The R-squared of the training dataset is 0.8927, and the R-squared for the testing dataset is 0.83273 based on Kaggle. We used a total of 5 predictors and 14 Betas.

Introduction:

Our goal is to create a regression model using the training dataset to predict the prices of cars in the testing dataset with high predictive power and the fewest predictor coefficients possible. We have been given two raw datasets: a training and a testing dataset. The training data set contains 23 variables with 1500 observations, while the testing dataset contains 22 variables with 938 observations. The testing dataset does not contain the variable PriceNew variable, which is why we will use the training dataset to create our model and then we will use the testing dataset to test how good our model is. In each of the datasets, there are 15 numerical variables (excluding PriceNew) and 7 categorical variables in both datasets. In our process of finding a regression model, we will be adjusting these datasets and adjusting some of these variables. The PriceNew variable in the training dataset is important as it tells us the prices of 1500 cars. The summary statistics of the variable are as follows:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
6945	14063	19188	21812	27206	65306

Methodology:

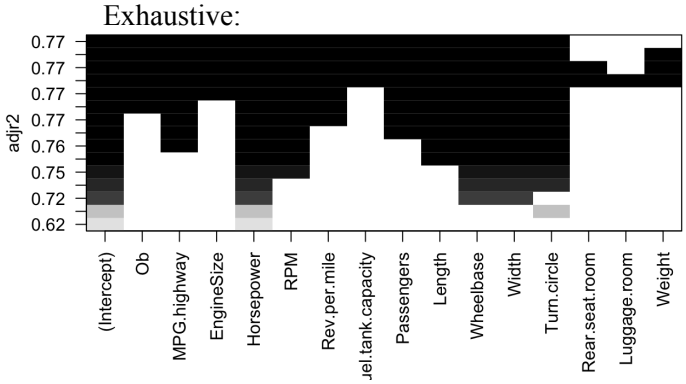
First, we are going to take a look at all the numeric variables and their correlations with "PriceNew."

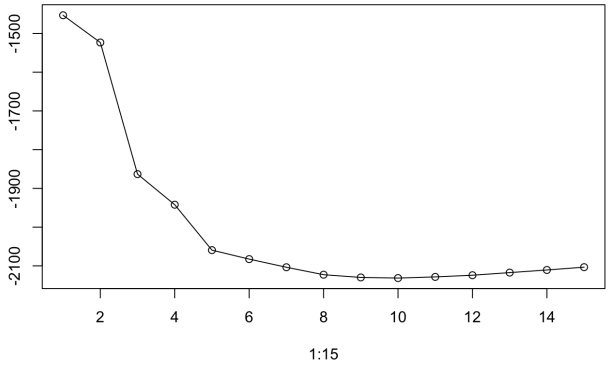
Ob	0.039
MPG.highway	-0.589
EngineSize	0.612
Horsepower	0.790
RPM	0.011
Rev.per.mile	-0.421
Fuel.tank.capacity	0.627
Passengers	0.131
Length	0.530
Wheelbase	0.563
Width	0.452

```
Turn.circle          0.369
Rear.seat.room       0.298
Luggage.room         0.264
Weight               0.663
PriceNew             1.000
```

We see that Horsepower, EngineSize, Fuel.tank.capacity, Length, MPG.highway are some of the numeric variables with the highest correlation. This gives us an idea of what predictors might be important to our model. We are going to run a regsubset using the leaps package in R to get the BIC and adjusted R-squared for forward, backward, and exhaustive selection. This will give us an idea of which variables to start with and how many predictors are optimal.

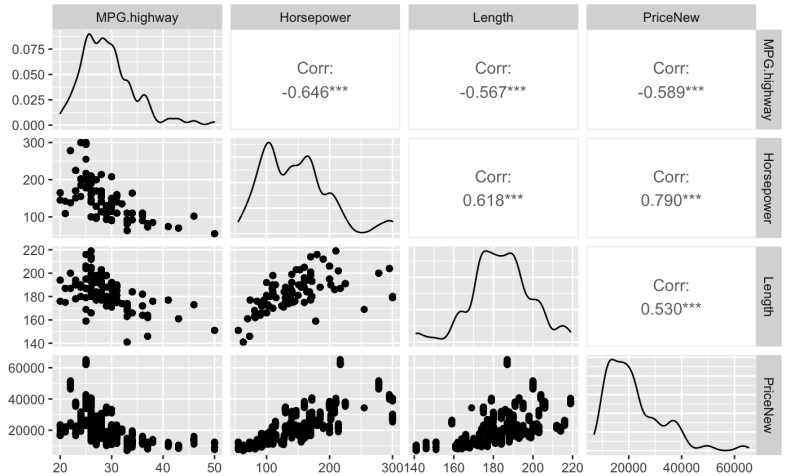
Exhaustive:





All of these give very similar results, so depicted is only the exhaustive method. With the best numerical predictors being MPG.highway, Horsepower, RPM, Rev.per.mile, Passengers, Length, Width, Turn.circle, and Wheelbase. We also see in the right graph the depiction of BIC, which is better when it is lower. It shows diminishing returns as you increase the number of predictors. This means that around 6-7 predictors would be great for what we are looking for since we don't want too many, as that will overcomplicate the model and maybe even run into the problem of overfitting. After taking a look at the categorical variables, the variable "Make" has an extremely high R-squared value to PriceNew, but since this variable has an enormous amount of coefficients, we will adjust this variable to make it more usable. The "Make" variable lists the model and the price of the car. We will where each car is from and categorize it into three regions: Europe, Asia, and the U.S. Our new predictor will be added to both the training and testing dataset and labeled as "MakeNew." The other predictors with good correlation to use are Type, AirBags, and Origin. Origin, however, would not be logical to use because it determines whether or not the car is USA-made, which our new predictor basically predicts. It would be pointless to use both variables as it overcomplicates the model without giving any more valuable information.

So our current model, after testing a large number of models, has predictors: Horsepower + MPG.highway + Length + MakeNew + AirBags + Type. Examining the numerical predictors, we can check for linearity. It looks like the predictors are linear to PriceNew, but we can test this later to confirm.

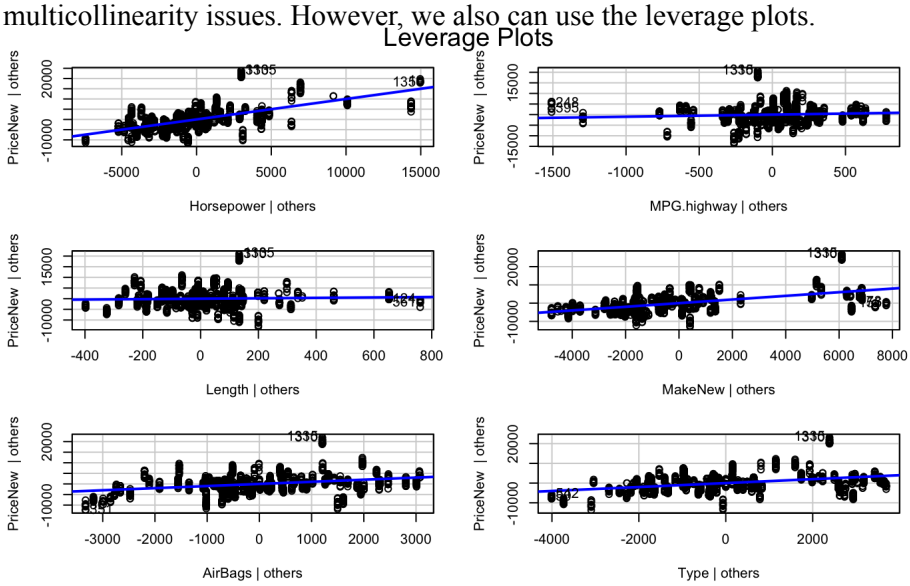


It does look like there could be some issues of multicollinearity. To be sure, we can check VIF in R studio and also the leverage plots.

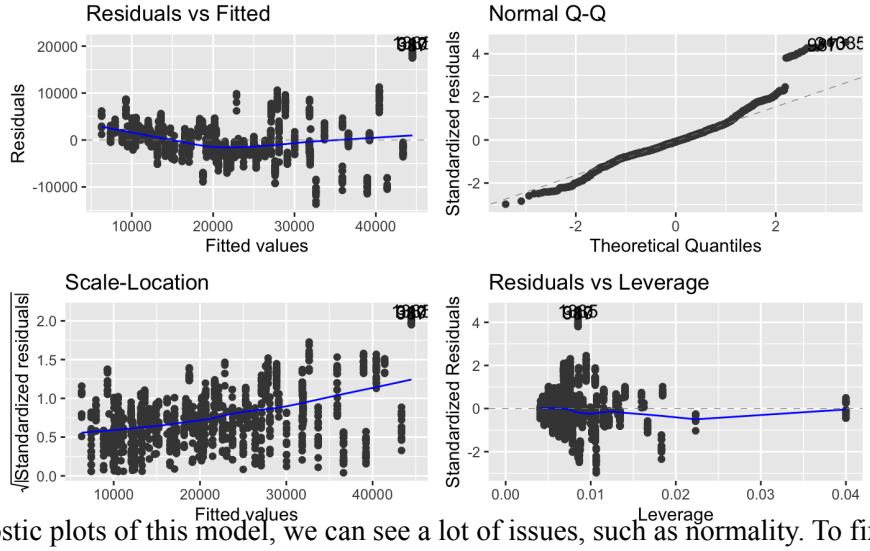
The VIF is as follows:

```
GVIF Df GVIF^(1/(2*Df))
Horsepower   2.636575  1        1.623753
MPG.highway  3.240303  1        1.800084
Length       4.141486  1        2.035064
MakeNew      2.025022  3        1.124790
AirBags      2.082224  2        1.201246
Type        13.077096  5        1.293157
```

None of the numeric variables are over 5, which means according to the VIF function in R, there aren't any serious multicollinearity issues. However, we also can use the leverage plots.

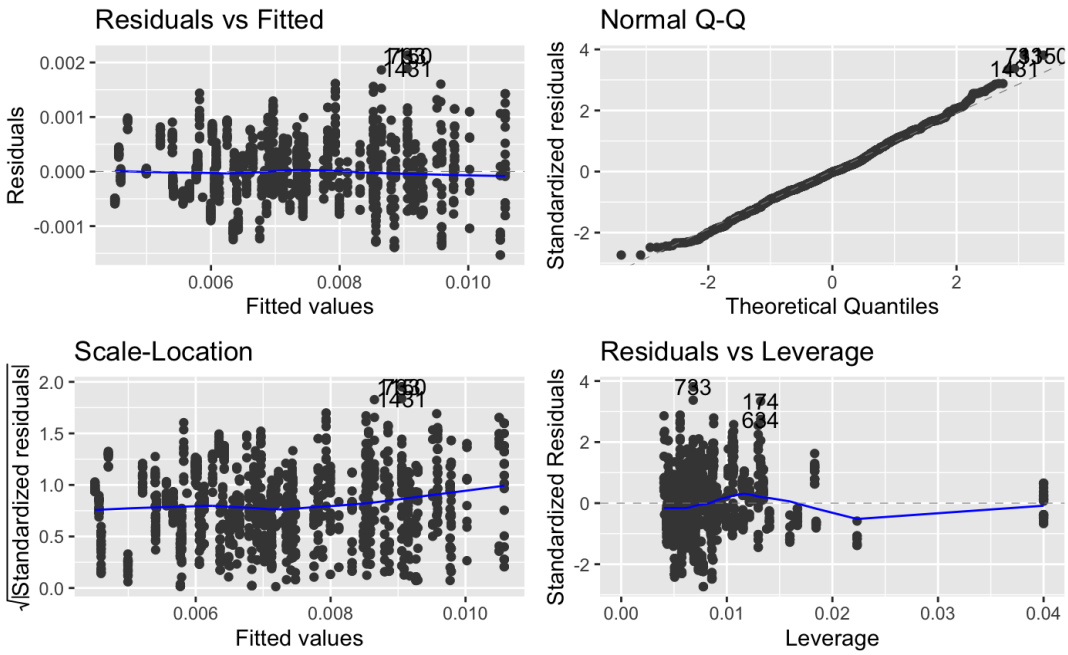


In the graphs here, we see that MPG.highway and Length have almost flat lines, which means that they are experiencing issues with multicollinearity. I suspect that they are linked with each other, so I will remove MPG.highway to see what happens. Our new model is now Horsepower + Length + MakeNew + AirBags + Type. We should also check the residuals and conditions of this model as well.

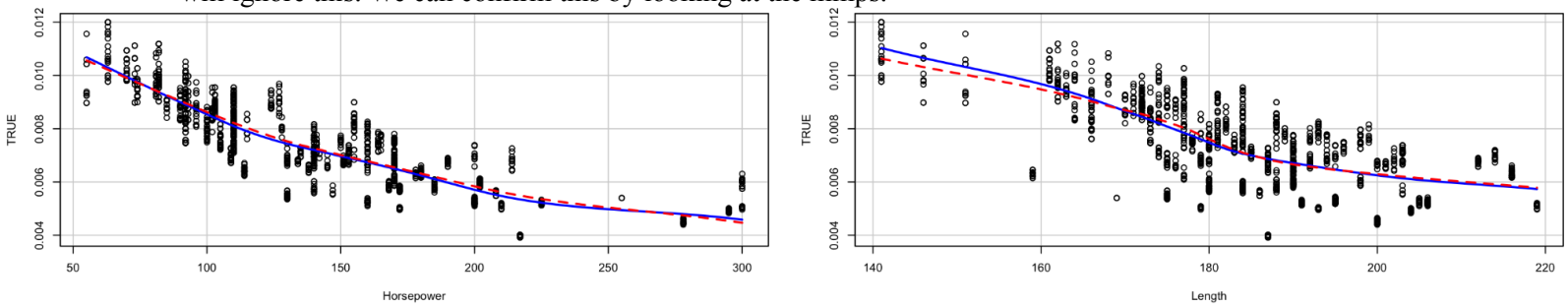


From the diagnostic plots of this model, we can see a lot of issues, such as normality. To fix this, we will transform using power transformation. In R, the power transformation gives us a lambda of -0.50, so we will use that for our model.

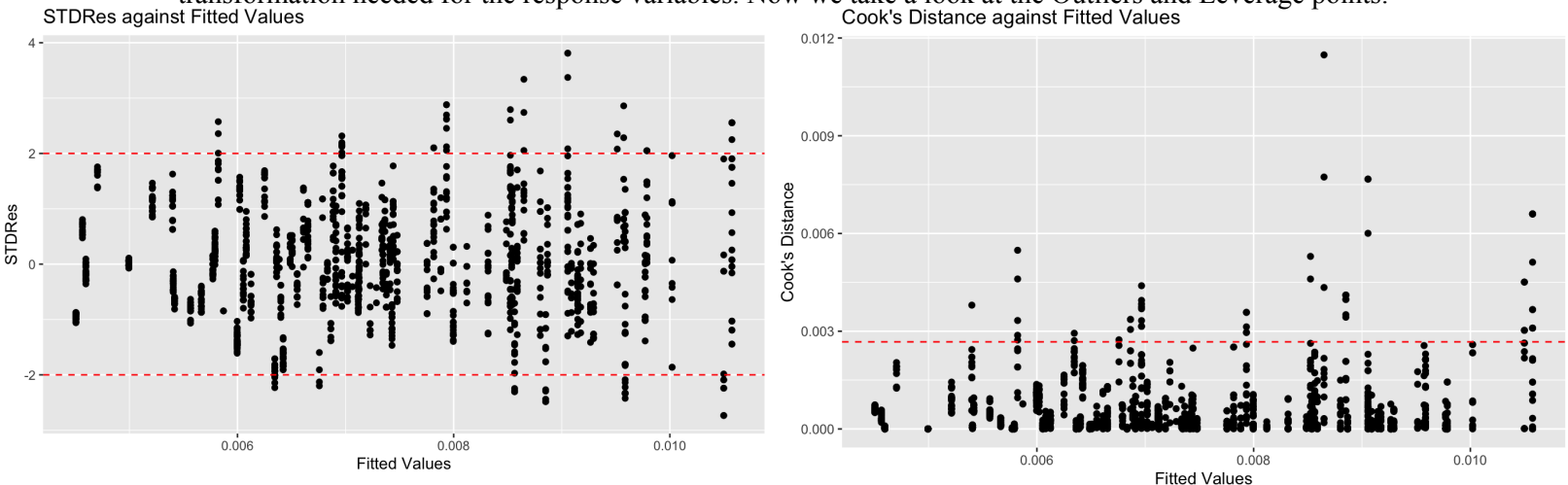
Our new model looks like this:
PriceNew^(-.5) ~ Horsepower + Length + MakeNew + AirBags + Type
This improves not only our diagnostics but also our R-squared, so this works very well.
Here are the diagnostics:



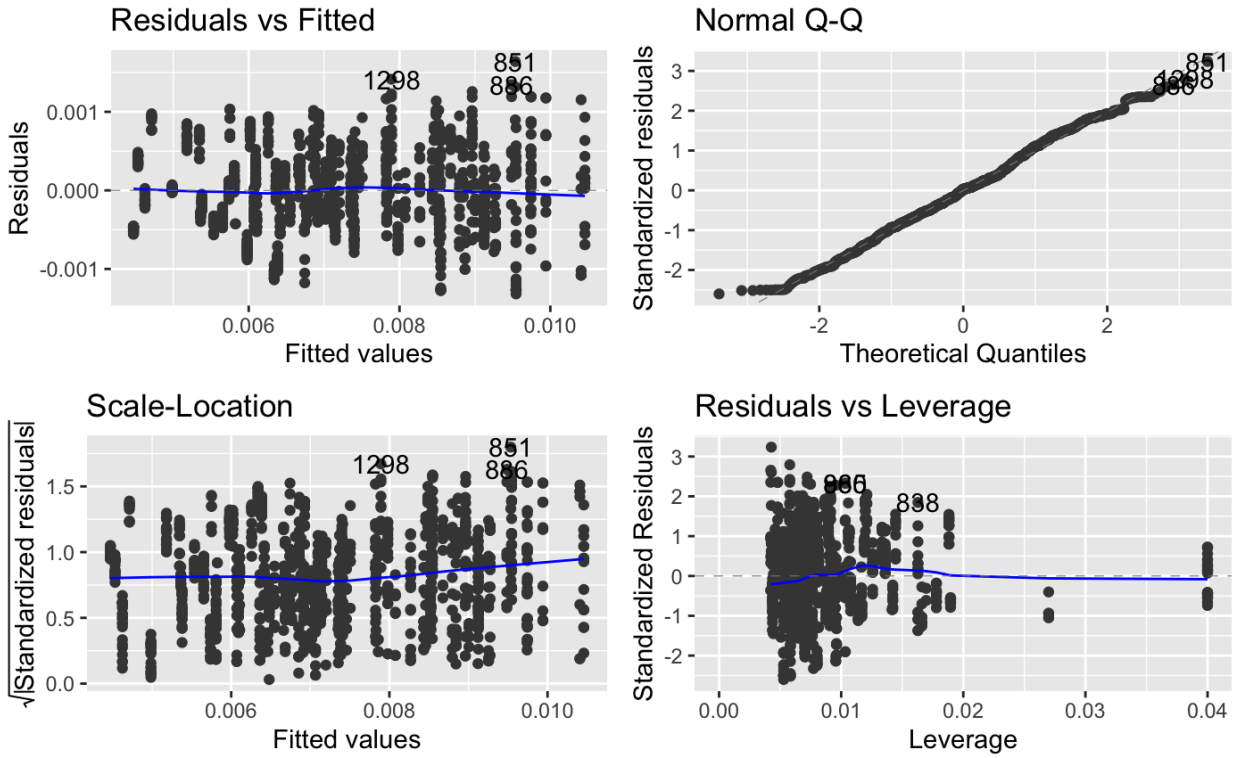
The diagnostics here look very good, except for some issues in the Residuals vs. Leverage plot. We will fix this after taking a look at the mmps. The power transformation gives us recommendations for the other variables and to transform them. However, since the other variables seem fairly linear to PriceNew, we will ignore this. We can confirm this by looking at the mmps.



We are looking for a separation in the red line and the blue line to indicate whether we need to transform any of the numerical predictors. It does not look like there are any substantial issues however an argument could be made about a slight separation in Length, but for our purposes, we shall say that there is no transformation needed for the response variables. Now we take a look at the Outliers and Leverage points.

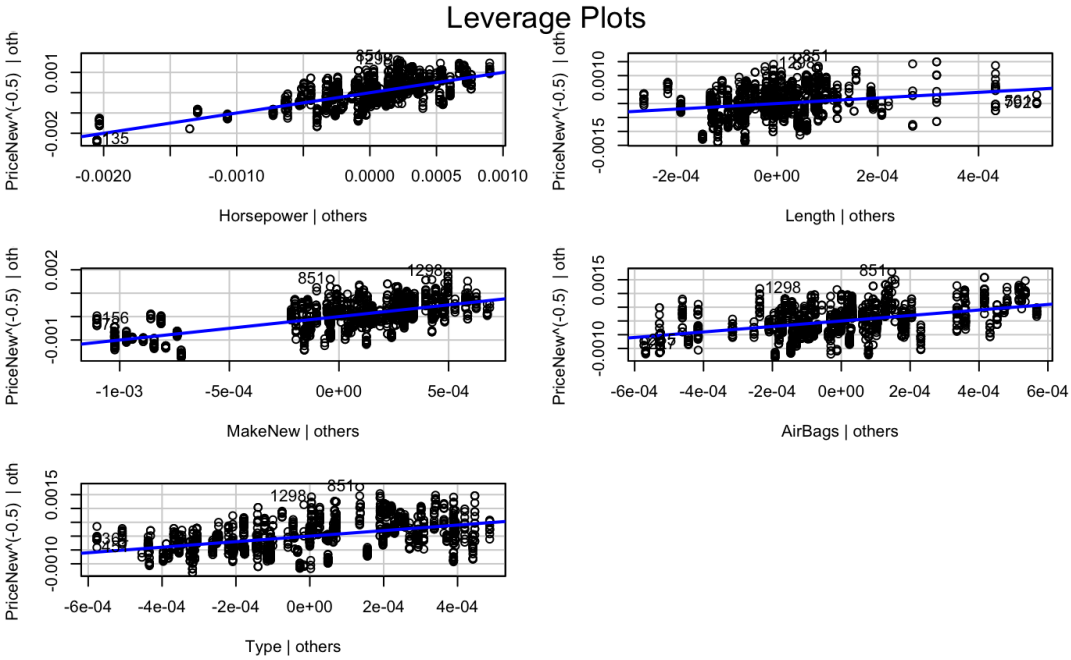


As we can see from the plots, there are some leverage points and outliers. We can remove the outliers and create a new dataset without the outliers. Our final model will be the same models and transformations but with the new data set, that has removed the outliers.
It is: PriceNew^(-.5) ~ Horsepower + Length + MakeNew + AirBags + Type
One last check of all the diagnostics:



VIF:

```
GVIF Df GVIF^(1/(2*Df))
Horsepower 2.143327  1        1.464011
Length     4.467865  1        2.113732
MakeNew    2.046549  3        1.126775
AirBags    2.111356  2        1.205425
Type       9.344348  5        1.250417
```



VIF looks much better than before; now, none of the predictors are flat lines. There appears to be some association still between the variables, but it is much better than before.

Results:
Our final model gives an adjusted R-squared of 0.8918 in R studio and 0.83273 on Kaggle. We used a total of 14 Betas and 5 predictors. The predictors we used were Horsepower, Length, MakeNew, AirBags, and Type.

Discussion:
Our findings indicated that the predictor's Horsepower, Length, MakeNew, AirBags, and Type can explain 0.8918 of the training data prices and 0.83273 of the testing data prices. The company “Geely Auto” can use our model to evaluate the Prices of their cars to send them out into the U.S. and European markets. There will, of course, be some inaccuracies with this model, but it is good for such few predictors.

Limitations and Conclusions:
There were a few limitations in our model. Firstly I wanted to get a much higher R-squared value, but there were many issues with this. At around 5 predictors, there was a diminishing return in adding more predictors and betas. Adding a few betas would only make a difference of 0.01 or so on the adjusted R-squared. Another thing I explored was interactions between variables. There were some interesting interactions between certain variables, but the improvement in R-squared was not worth the number of betas the interaction was creating. Adding an interaction of any of the categorical variables would make the betas increase by so much, so this is the reason I decided not to put any interactions in my model. Additionally, for points with high leverage, I had no choice but to remove them or combine them with another variable. At first, combining variables seemed promising, but the combined variables I tested either ended up with a worse multicollinearity problem or just did not help the R-squared at all. This is why I could only remove the variables that had high multicollinearity with others.