Optimizing Auction Price Predictions for BMW Vehicles: An Analysis of Linear, Lasso, and Ridge Regression Models

Lab C2 Group 1
Grant Mason Gealy, Seungsoo Lee, Virushi Nishith Patel, Yan Si, Caiwei Zhang, Xiang Zhao

Abstract

The goal of this report is to develop a predictive linear model for BMW auction strike prices using a dataset split into training and validation sets. Our primary objectives include identifying key vehicle features for price prediction, building a robust predictive model through stepwise regression, and investigating the relationship between price and car age. The final model, applied to the validation set, achieves an adjusted R-squared of 0.82 and an RMSE of 3604.34. Additional exploration with Lasso and Ridge regression models yields a slight improvement in RMSE for Lasso to 3589.90, but no enhancement in adjusted R-squared. While the model demonstrates proficiency in predicting mid-range prices, there is an opportunity for refinement, especially in enhancing precision for vehicles with low prices.

Introduction

This report aims to establish an understanding of BMW car auction prices. Our primary goal is to find impactful vehicle features, then construct a reliable model to predict the vehicle auction price. We are also interested in finding the relationship between vehicle price and car age, which counts the number of days between car registration date and car sales date. Our analysis was conducted in two steps. First, we trained a model to best predict price with the training set. Next, we applied the selected prediction model to predict the price in the validation set and assess the model performance. We further computed a Lasso and Ridge model to see if it would improve our prediction performance. We highlighted the strengths of our models and discussed the limitations of our model in the final report.

Data

The BMW Pricing Challenge dataset consists of price and vehicle information of 4,843 BMW auctioned in 2018. Price is a continuous variable defined as the highest bid in U.S. dollars. Information about each sold BMW car includes continuous variables mileage and engine power, and categorical variables model key, car type, paint color, fuel type, registration date, sold date, and eight dichotomously coded unknown features. The dataset was split into a training set (n=2,434) and a validation set (n=2,409). Across the training and validation set, one observation with negative mileage and another observation with zero engine power were excluded from further analysis, ending up with a final training sample size of 2,433 and a final validation sample size of 2,408. We took log transformation to account for the right skewness in the raw price variable. We further created two new variables: car age and total number of features. Car age is defined as the difference in days between the sold date and the registration date. The total number of features represents the number of features a car has out of eight features. Additionally, the variable model key records 75 different exact BMW models. To ensure an adequate number of observations in each level exist in both the training and validation set and avoid model overfitting, we categorized the BMW models into different series. For example, models X1, X3, X4, X5, and X6 were categorized into a group called X series. In the end, we had 11 levels for the variable model key, which is different from the model in Report 3. We also did not group the categorical variables (paint color, car type, and fuel type) to better capture the variability in price and a more accurate prediction. Summary statistics of each variable in the training and validation sets are displayed in detail in Table 1. Some additional insight from the dataset was that for the training set, there were 22 cars priced below $1,000, while in the validation set, there were 40 cars priced below $1,000.

Modeling and Analysis for the Full Data Set

Project Report 3 primarily focused on delineating which vehicle features impact price and finding the relationship between vehicle price and car age using the full dataset. Two distinct linear regression models are constructed to address each question, incorporating forward stepwise selection to identify influential predictors. In the context of the first question, the model reveals a moderately good fit with an adjusted R-squared value of 0.534. Notably, engine power and mileage emerge as pivotal determinants, showcasing a positive association between engine power and log(price) and a negative association between mileage and log(price). Six features (features 1-6) exhibit noteworthy impacts on prices, each contributing to a percentage increase in log(price). Additionally, model types, specifically 5 Series and X Series, exert a considerable influence on pricing compared to the baseline 3 Series. The main findings of this model identified engine power, mileage, and specific features (1, 2, 4, 5, 6, 8) as significant predictors of log-transformed vehicle prices.

The second question explores the impact of car age on log(price), and the resulting model attains a moderately good fit with an adjusted R-squared value of 0.6893. Engine power and mileage maintain their significance, with engine power displaying a positive association and mileage exhibiting a negative correlation with log(price). Features 1, 2, 4, 5, 6, and 8 continue to exert substantial effects on prices, contributing to percentage increases in log(price). Model types, particularly 5 Series and X Series still emerge as influential factors, showcasing their impact on pricing relative to the 3 Series. The main finding of this model is that including car age in the model increased the explanatory power to 67%. Car age emerged as a significant factor, with each additional year leading to about 11% decrease in raw price (0.032% decrease in raw price per day).

Modeling and Analysis for the Training Data Set

By examining various transformations and categorizations, we attempted to build the best predictive model for BMW car prices. We successfully reached an adjusted R-squared of 0.75, demonstrating a significant amount of explanatory power. The addition of a feature total variable, which combined the overall influence of all the separate features, aided in prediction. Unlike previous studies that concentrated on marginal effects, the modeling technique stressed the need of examining different transformations and groupings to boost forecast accuracy. In order to maximize prediction accuracy, complicated models, log transformations, and variable groups were examined. Even though the adjusted R-squared of 0.75 represents a noteworthy improvement, further improvement is still required, particularly for forecasts of extremely high values. The training and validation datasets demonstrated good consistency in categorical and binary features, crucial for model generalizability. However, discrepancies in numerical features like engine power and price could impact model performance. The scatterplot analysis of the training dataset revealed a robust correlation between predicted and actual values, particularly for mid-range data points, indicating accuracy. Yet, variability and outliers in higher value ranges signaled areas for improvement, aligning with the validation dataset's findings. Quantitatively, the model exhibited a solid performance with an R-squared of 0.8183, explaining 81.86% of variance. Despite this, moderate prediction errors, as indicated by RMSE and MAE, and a MAPE of 53.86%, emphasized areas for potential refinement in the model's precision. The mathematical equation of the model and its assumptions are as follows:

$$\begin{aligned}\log(Price_i) = {} & \beta_0 + \beta_1 Car\ Age_i + \beta_2 Engine\ Power_i + \beta_3 Car\ type_i + \beta_4 Total\ \#\ features_i + \beta_5 Mileage_i \\ & + \beta_6 Model\ key_i + \beta_7 Paint\ color_i + \beta_8 Fuel_i + \beta_9 Feature\ 3_i + \beta_{10} Feature\ 6_i \\ & + \beta_{11} Feature\ 5_i + \beta_{12} Feature\ 4_i + \epsilon_i,\end{aligned}$$

The assumptions are: 1) a linear relationship between the dependent and independent variables; 2) The error terms have 0 mean i.e., $E(\epsilon_i) = 0$; 3) The error terms have equal variance i.e., $Var(\epsilon_i) = \sigma^2$; 4) The error terms are normally distributed i.e., $\epsilon_i \sim N(0, \sigma^2)$; and 5) The error terms $\epsilon_1, \dots, \epsilon_n$ are independent.

The following interpretations of coefficients are considered at level α =0.05 while holding other covariates constant. A brand new, beige, 1-series, convertible car with 0 engine power and 0 features is estimated to be priced at 9.78 on log scale (≈ \$17676.65). Continuous variables car age, engine power, total number of features, and mileage are significantly associated with log(price). Respectively, a one-unit increase in car age, engine power, total number of features, and mileage is associated with a 0.0003-unit decrease, 0.004-unit increase, 0.097-unit increase, and 1.9×10^{-6} unit decrease in log(price) (≈ 0.03% decrease, 0.4% increase, 10% increase, and 0.0002% decrease in raw price). Categorical variables were associated with log(price) at different levels. Compared to the price of convertible cars, prices of coupe cars and SUVs were not significantly different. On average, estate, hatchback, sedan, subcompact, and van cars were priced significantly lower by 0.39, 0.22, 0.21, 0.18, and 0.39 units on the log scale (≈ 33 %, 20%, 19%, 16 %, and 32% decrease in raw price) compared to convertible cars. Compared to 1- series cars, 3, 4, 5, 6, 7, i, M, and Z series cars were priced significantly higher by 0.12, 0.18, 0.23, 0.32, 0.32, 0.70, 0.20, and 0.37 units on the log scale (≈ 13%, 20%, 26%, 38%, 38%, 101%, 22%, and 45% increase in raw price) while 2 and X series cars had no significant difference in price. Compared to beige cars, only blue and green cars were priced significantly lower by 0.14 and 0.35 units on the log scale (≈ 13%, and 30% decrease in raw price). The rest of the colors had no significant associations with log(price). Compared to diesel cars, electro and hybrid petrol cars had no significant difference in log(price), while petrol cars were priced significantly lower by 0.12 units on the log scale. Additionally, features 3, 6, and 5 were significantly associated with a decrease in log(price) by 0.06, 0.04, and 0.04 units (6%, 4%, and 4% decrease in raw price), respectively. Feature 4 was not significantly associated with price.

Modeling and analysis of LASSO and ridge regression in the training data

We also implemented LASSO and ridge regression in the training data. We set a full model that consisted of 15 predictors: car age, engine power, car type, total number of features, mileage, model key, paint color, fuel, features 1-6, and 8. Both LASSO and ridge regressions treat each level of a categorical variable as an individual binary indicator, resulting in an expansion from 15 to 40 input variables (excluding the intercept) into the algorithm. Utilizing a 10-fold cross-validation, the LASSO model yielded the best lambda at 0.000406 and restricted coefficients for five variables to be 0 (Table 2). The ridge regression resulted in the best lambda at 0.0396. The coefficient estimates are displayed in Table 2.

Prediction using Ordinary Least Square regression in the validation data

Since we filtered the validation dataset and training dataset by *obs_type* of “Validation” and “Training” in the full dataset, these two datasets are mutually exclusive and collectively exhaustive. The means and standard deviations for numerical variables in training and validation sets are similar, except for a noticeable difference in the average price (Table 1). A Welch Two Sample t-test on the log-transformed prices showed a small yet significant difference, with means of 9.5246 and 9.4876 in the training and validation datasets, respectively (p-value = 0.0473). The distributions of categorical and binary features (feature 1 to 8) are consistent across both datasets, though some discrepancies in model key are noted.

Using our final model, we predicted log(price) and exponentiated it to raw scale to perform a series of predictive metrics. The prediction yielded an adjusted R-squared of 0.82, indicating our final prediction model explains about 82% of the variance in the validation data. The Root Mean Square Error (RMSE) is at 3604.34 and the Mean Absolute Error (MAE) is at 2359.25, which points to moderate prediction errors. The Mean Absolute Percentage Error (MAPE) of 53.86% shows the average percentage deviation of the predictions from actual values, suggesting room for improvement. The model shows strong correlation for mid-range data points but less precision at higher value ranges (Figure 5), overpredicting in low-range and underpredicting in high-range price segments. In the validation dataset, the scatterplot showed a commendable alignment of predictions with actual values for mid-priced vehicles, while there was a noticeable increase in prediction scatter for lower-priced vehicles. A significant trend of overprediction was observed in the low-range price segment, further suggesting areas where the model's accuracy could be enhanced.

Prediction using LASSO and ridge regression in the validation data

We applied the best LASSO model and ridge regression to the validation set for prediction. The resulting prediction accuracy metrics are included in Table 3. Compared to our prediction model selected via stepwise methods in Repot 4, the LASSO model yielded a better prediction accuracy while the ridge regression made a worse prediction ($RMSE_{LASSO}$=3589.90 < $RMSE_{OLS}$ =3604.34 < $RMSE_{ridge}$=3611.29) in terms of RMSE. All of them resulted in similar adjusted R-squared (adjusted R-squared =0.82) in the validation set.

Discussion

In this study, we identified determinants of auction price variation in 4841 BMW vehicles. As was expected, engine power was positively associated with price while mileage was negatively associated with price. Though the original dataset did not provide car age, we created it and identified that car age was the most significant predictor associated with price. In addition, 5 series cars and X series were priced significantly higher than 3 series cars and other models. Among the eight unknown features, features 1, 2, 4, 5, 6, and 8 were significantly and positively associated with price. Splitting the full dataset into training and validation, we were able to train prediction models using stepwise methods, LASSO, and ridge regression in the training dataset and make accurate predictions in the validation set. The optimal LASSO model yielded the best prediction (RMSE=3589) followed by the OLS model selected via stepwise regression (RMSE=3604) and the optimal ridge regression (RMSE=3611).

Our results echoed closely with our goals set at the beginning of the semester. First, we successfully identified the factors related to car prices via linear models. Second, we defined the variable car age, modeled its association with price, and concluded that there was a significant linear association between them. Third, based on the given information, we successfully trained prediction models and achieved high accuracy when tested in the validation set.

Our study should be considered with several limitations. First, we did not examine any interaction terms in either inference models or prediction models. In reality, a specific car model might be more popular in a certain color and a particular car type might depreciate less over time. Thus, including meaningful interaction terms could contribute to the evaluation of effect modification and prediction. Second, though we reached a high adjusted R-squared in the validation set, none of the prediction models accurately predicted the prices of cars with low values. Further analysis should seek to search for other vital car information like the maintenance history and number of accidents that are closely related to price. Third, our analysis did not account for the dynamic influence of time on price trends. Additional analyses could be performed to identify if there is a seasonal pattern in price. Such patterns could also reflect latent information such as tax returns or industrial promotion. Lastly, collecting more data would be beneficial to build a more predictive model and make more reliable inferences.

Author Contribution Statement:

GMG: coding (statistical analysis in report 2), writing (data analysis in report 3, modeling and analysis in report 4, prediction in final report).
SL: coding (data preprocessing, visualization of summary data in report 2, 3, 4, prediction in report 3), writing (introduction, and revisions report 2, 3).
VNP: interpreting results and writing (report 2), statistical analysis coding and writing data analysis (report 3), coding data processing, visualization, statistical modeling, prediction, and writing (report 4), writing modeling and analysis for the training data set (final report).
YS: coding (data processing, visualization, statistical modeling, prediction in report 4), writing (result interpretation in report 2, data analysis in report 3, prediction in report 4, abstract and introduction in final report), reviewing.
CZ: coding (data processing, visualization, and statistical analysis in report 2), writing (questions in report 1, statistical modeling in report 3, prediction in report 4, modeling and analysis in final report), reviewing.
XZ: project coordination, conception, and design (report 1), coding (data processing and visualization in reports 2-4 and final report; statistical analysis in reports 3, 4, and final report; prediction in report 4 and final report), writing (modeling and analysis in report 4, and final report; prediction in final report; reviewing and editing in reports 1-4 and final report).

Table 1. Summary statistics of the BMW pricing data by training and validation

Variable	Mean (SD) or N (%)		Variable	N(%)	
	Training	Validation		Training	Validation
Price (dollars)	16036.5 (9844.43)	15594.64 (8457.94)	3 series	885 (36.37%)	915 (38.00%)
Log (Price)	9.52 (0.62)	9.49 (0.68)	5 series	587 (24.13%)	553 (22.97%)
Mileage (miles)	141004.20 (60169.01)	141420.36 (61901.36)	X series	534 (21.95%)	527 (21.89%)
Engine power (horsepower)	128.99 (38.93)	128.06 (37.52)	Other series (1, 2, 4, 6, 7, i, M, and Z)	427 (17.55%)	427 (17.55%)
Car age (days)	2016.34 (922.45)	2023.81 (923.25)	Black	814 (33.46%)	818 (33.97%)
Total number of features	3.94 (1.87)	3.9 (1.9)	Grey	588 (24.17%)	587 (24.38%)
Feature 1 (yes)	1362 (55.98%)	1299 (53.95%)	Others (white, blue, silver, brown, green…)	1031 (42.38%)	1003 (41.65%)
Feature 2 (yes)	1958 (80.48%)	1880 (78.07%)	Estate	802 (32.96%))	804 (33.39%)
Feature 3 (yes)	509 (20.92%)	469 (19.48%)	Sedan	583 (23.96%)	584 (24.25%)
Feature 4 (yes)	480 (19.73%)	481 (19.98%)	SUV	532 (21.87%)	525 (21.80%)
Feature 5 (yes)	1101 (45.25%)	1129 (46.89%)	Others (coupe, van, convertible, hatchback…)	516 (20.21%)	495 (20.56%)
Feature 6 (yes)	576 (23.67%)	593 (24.63%)	Diesel	2324 (95.52%)	2315 (96.14%)
Feature 7 (yes)	2280 (93.71%)	2233 (92.73%)	Other fuels (electro, petrol, and hybrid petrol)	109 (4.48%)	93 (3.86%)
Feature 8 (yes)	1312 (53.93%)	1307 (54.28%)			

Table 2. Coefficient estimates from the final OLS model, LASSO model, and ridge regression

Variable	β_{OLS}	β_{LASSO}	β_{ridge}	Variable	β_{OLS}	β_{LASSO}	β_{ridge}	Variable	β_{OLS}	β_{LASSO}	β_{ridge}
Intercept	9.78 ***	9.66	9.59	4 Series	0.18 **	0.17	0.15	Red	-0.05	0.01	0.03
Car Age	-3.32×10^{-4} ***	-3.30×10^{-4}	-3.03×10^{-4}	5 Series	0.23 ***	0.21	0.12	Silver	-0.08	-0.02	-0.02
Engine Power	0.004 ***	0.004	0.004	6 Series	0.32 ***	0.30	0.27	White	-0.09	-0.03	-0.02
Coupe	-0.13	-0.05	0.06	7 Series	0.32 ***	0.30	0.23	Electro	-0.38	-0.30	-0.19
Estate	-0.39 ***	-0.31	-0.15	i Series	0.70 **	0.64	0.50	Hybrid Petrol	-0.03	0	0.06
Hatchback	-0.22 **	-0.14	-0.04	M Series	0.20 *	0.17	0.17	Petrol	-0.12 ***	-0.12	-0.12
Sedan	-0.21 **	-0.13	0.01	X Series	0.21	0.12	0.09	Feature 1	NA	0	0.06
Subcompact	-0.18 *	-0.11	-0.03	Z Series	0.37 *	0.41	0.43	Feature 2	NA	0.01	0.09
SUV	-0.16	0	0.09	Black	-0.10	0	0.01	Feature 3	-0.06 **	-0.05	0.01
Van	-0.39 **	-0.28	-0.19	Blue	-0.14 *	-0.08	-0.08	Feature 4	-0.04 *	-0.03	0.04
Total # Features	0.10 ***	0.09	0.03	Brown	-0.07	-0.01	-4.62×10^{-4}	Feature 5	-0.04 *	-0.03	0.04
Mileage	-1.90×10^{-6} ***	-1.89×10^{-6}	-1.93×10^{-6}	Green	-0.35 **	-0.29	-0.33	Feature 6	-0.04	-0.03	0.03
2 Series	0.03	0	-4.20×10^{-5}	Grey	-0.07	-0.01	-0.01	Feature 8	NA	0.01	0.08
3 Series	0.12 ***	0.10	0.02	Orange	-0.23	-0.13	-0.08				

Note: *** $p<0.001$; ** $p<0.01$; * $p<0.05$.

Table 3. Prediction accuracy in the validation set for the model trained by OLS, LASSO, and ridge regression

Model	RMSE	MAE	MAPE	Adj-R^2
OLS	3604.34	2359.25	0.54	0.82
LASSO	3589.90	2345.03	0.54	0.82
Ridge Regression	3611.29	2345.48	0.54	0.82

Figure 1. Log(price) vs continuous variables in the training data

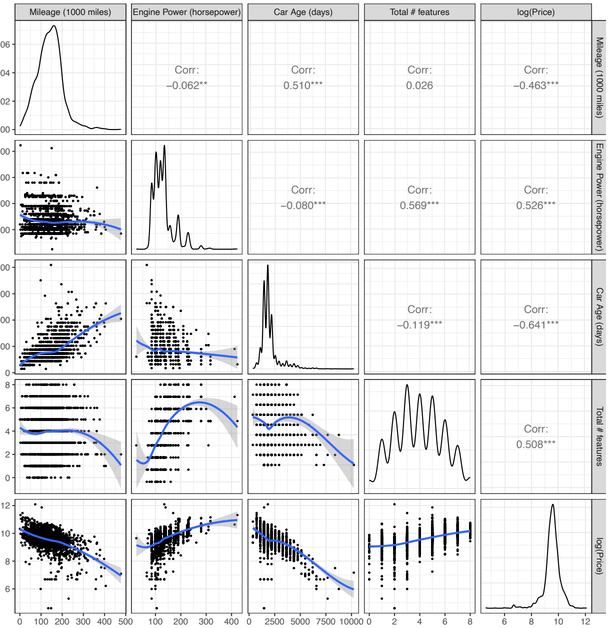


Figure 2. Log(price) vs categorical variables in the training data

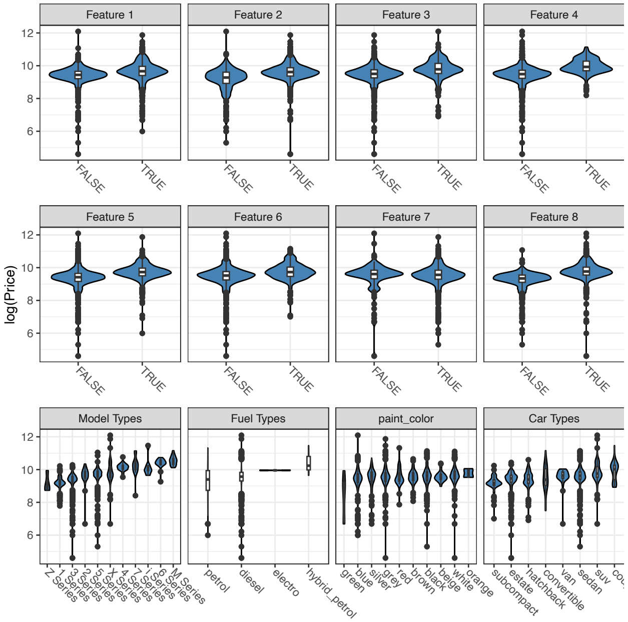


Figure 3. Residual diagnostic plots of the final OLS model fit

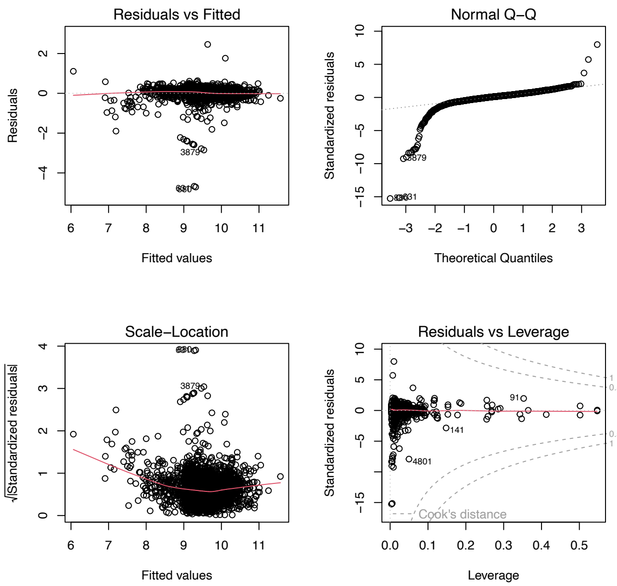


Figure 4. Coefficient path vs λ for the LASSO/ridge regression

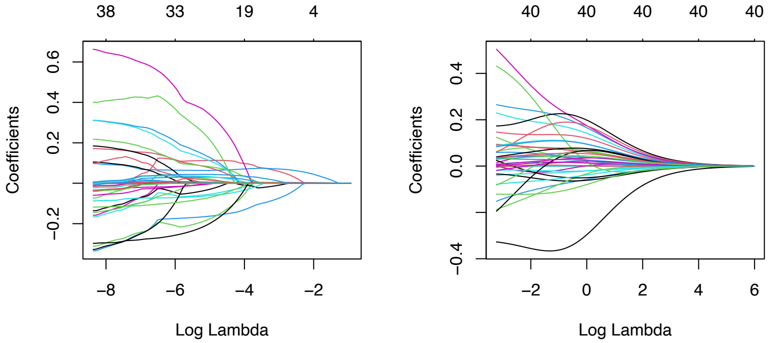
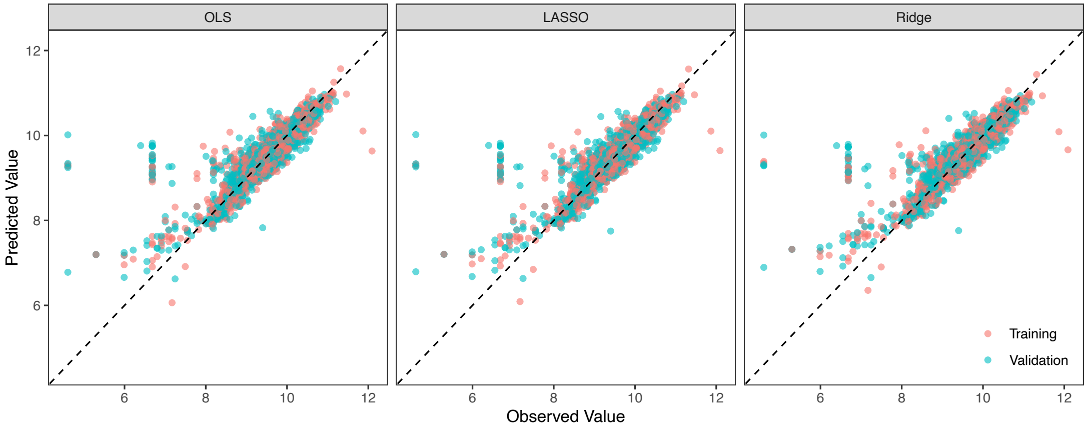


Figure 5. Predicted vs observed log(price) by OLS, LASSO, and ridge regression



Appendix:

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(glmnet)
library(nlme)
library(dplyr)
library(car)
library(MLmetrics)
library(GGally)
## load data
path <- "/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/Report4/"
bmw <- read.csv(paste0(path,"BMWpricing_updated.csv"))
table(bmw$obs_type)
bmw%>%filter(mileage<0 | engine_power==0)
## generate a new df excluding the obs with negative mileage and 0 engine power
bmw <- bmw%>%filter(mileage>0 & engine_power!=0)
## generate log(price) and lasting time
## take log transformation on price
bmw$price_log <- log(bmw$price)

# calculate the difference in days
bmw$registration_date <- as.Date(bmw$registration_date, format="%d/%m/%Y")
bmw$sold_at <- as.Date(bmw$sold_at, format="%d/%m/%Y")
bmw$car_age <- as.numeric(difftime(bmw$sold_at,
                                   bmw$registration_date,
                                   units = "days"))
## generate a numerical feature variable
bmw$feature_sum <- rowSums(bmw[,paste0("feature_",1:8)])

## recategorize model key (75 levels to 11 levels)
bmw2 <- bmw%>%
  mutate(model_key=paste(substr(gsub("ActiveHybrid ","",model_key),1,1),"Series")
         # model_key=ifelse(model_key%in%c("X Series","5 Series","3 Series"),model_key,"others"),
         # fuel=ifelse(fuel=="diesel",fuel,"others"),
         # paint_color=ifelse(paint_color%in%c("black","grey"),paint_color,"others"),
         # car_type=ifelse(car_type%in%c("suv","sedan","estate"),car_type,"others")
  )
training_data <- subset(bmw2, obs_type == "Training")
validation_data <- subset(bmw2, obs_type == "Validation")

sum(is.na(training_data))
sum(is.na(validation_data))

## generate a matrix for prediction in the validation set
val.mat <- model.matrix(lm(formula = rep(1,dim(validation_data)[1]) ~ car_age +
                             engine_power + car_type + feature_sum +
                             mileage + model_key + paint_color + fuel +
                             feature_1 + feature_2 + feature_3 + feature_4 +
                             feature_5 + feature_6 + feature_8,
                           data = validation_data))
cont_vars <- c("mileage","engine_power","car_age","feature_sum","price_log")
my_fn <- function(data, mapping, method="loess", ...){
      p <- ggplot(data = data, mapping = mapping) +
      geom_point(size = 0.5) +
      geom_smooth(method=method, ...)
      p
}
ggpairs(training_data[,cont_vars], lower = list(continuous = my_fn, scale = "free"),
        progress = FALSE) + theme_bw()
ggpairs(validation_data[,cont_vars], lower = list(continuous = my_fn, scale = "free"),
        progress = FALSE) + theme_bw()
ggpairdata <- training_data[,cont_vars]%>%
mutate(mileage = mileage/1000)%>%
rename(`Mileage (1000 miles)` = "mileage",
       `Engine Power (horsepower)` = "engine_power",
      `Car Age (days)` = "car_age",
      `Total # features` = "feature_sum",
      `log(Price)` = "price_log")
pdf("/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/Report4/Figure1.pdf", height = 8, width = 8)
ggpairs(ggpairdata, lower = list(continuous = my_fn, scale = "free")) + theme_bw()
dev.off()
cate_vars <- c(paste0(c("feature_"),1:8),"model_key","fuel","paint_color","car_type")
training_cate_long <- training_data[,c("price_log",cate_vars)]%>%
    rename(`Feature 1` = "feature_1",`Feature 2` = "feature_2",
            `Feature 3` = "feature_3",`Feature 4` = "feature_4",
            `Feature 5` = "feature_5",`Feature 6` = "feature_6",
            `Feature 7` = "feature_7",`Feature 8` = "feature_8",
            `Model Types` = "model_key",`Car Types` = "car_type",`Fuel Types`="fuel")%>%
    reshape2::melt(.,id.vars = c("price_log"),variable.name = "CarInfo", value.name = "Value")

## Generate violin plots of Price vs fuel, paint color, and car type as Figure 2
pdf("/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/Report4/Figure2.pdf")
ggplot(data = training_cate_long) + aes(x=reorder(Value,price_log), y = price_log) +
  geom_violin(color="black", fill = "steelblue") +
  geom_boxplot(width=0.1, fill="white") +
  labs(x=NULL, y="log(Price)") +
  facet_wrap(~CarInfo, nrow=3, scales = "free_x") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 315, vjust = 0.5, hjust=0))
dev.off()
# prediction model used in report 4
model.list <- list("price_log ~ car_age + engine_power + feature_sum + mileage +
          model_key + car_type + fuel + paint_color + feature_1 + feature_2 +
          feature_3 + feature_4 + feature_5 + feature_6 + feature_8",
          ## Model 1: Full model with both continuous and categorical
          ## feature variables (omitting feature 7 to aviod complete multicollinearity)
          "price_log ~ car_age + engine_power + feature_sum + mileage +
          model_key + car_type + fuel + paint_color + feature_1 + feature_2 +
          feature_3 + feature_4 + feature_5 + feature_6 + feature_8 + I(engine_power^2)"
          ## Model 7: Model 1 adding mileage^2 and engine power^2
          ## based on the scatterplot matrix
          )
step_result <- data.frame()
## loop over all combinations
for (j in 1:length(model.list)) {
    ## fit a full model first
    fit.full <- lm(model.list[[j]], data = training_data)
    ## variable selection using stepwise regressions
    for.aic <- step(lm(price_log ~ 1, data = training_data), direction = "forward",

    scope = formula(fit.full), trace = 0) # forward AIC

    for.bic <- step(lm(price_log ~ 1, data = training_data), direction = "forward",
    scope = formula(fit.full), k = log(2433), trace = 0) # forward BIC
    back.aic <- step(fit.full, direction = "backward", trace = 0) # backward AIC
    back.bic <- step(fit.full, direction = "backward", k = log(2433), trace = 0) # backward BIC
    step.aic <- step(lm(price_log ~ 1, data = training_data), direction = "both",
                    scope = formula(fit.full), trace = 0) # stepwise AIC
    step.bic <- step(lm(price_log ~ 1, data = training_data), direction = "both",
    scope = formula(fit.full), k = log(2433), trace = 0) # stepwise BIC

    Adjusted_R.square <- data.frame("Method"=c("for.aic", "for.bic",
    "back.aic", "back.bic",
    "step.aic","step.bic"),

    "Adj.r.square"=c(summary(for.aic)$adj.r.square,
    summary(for.bic)$adj.r.square,
    summary(back.aic)$adj.r.square,
    summary(back.bic)$adj.r.square,
    summary(step.aic)$adj.r.square,
    summary(step.bic)$adj.r.square))
    best_model <- get(Adjusted_R.square$Method[which.max(Adjusted_R.square$Adj.r.square)])
    temp <- data.frame(Data = "bmw2",

    Predictors = paste(colnames(best_model$model)[-1],collapse = "/"),
    Model = deparse1(best_model$call),
    AIC=AIC(best_model),
    BIC=BIC(best_model),
    Adj.r.square=summary(best_model)$adj.r.square)

    step_result <- rbind(step_result,temp)
}
which.max(step_result$Adj.r.square)
which.min(step_result$AIC)
step_result[which.max(step_result$Adj.r.square),]
best_model <- lm(formula = price_log ~ car_age + engine_power + car_type + feature_sum +
          mileage + model_key + paint_color + fuel + feature_3 + feature_6 +
          feature_5 + feature_4 + I(engine_power^2), data = training_data)

vif(best_model)
step_result2 <- step_result[grep("I\\(engine\\_power\\^2\\)",step_result$Model,invert = T),]
step_result2
final_model <- lm(formula = price_log ~ car_age + engine_power + car_type + feature_sum +
          mileage + model_key + paint_color + fuel + feature_3 + feature_6 +
          feature_5 + feature_4, data = training_data)

vif(final_model)
anova(final_model, best_model)
```

```r
pdf("/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/Report4/Figure3.pdf")
par(mfrow=c(2,2))
plot(final_model)
dev.off()
summary(final_model)
write.csv(data.frame(summary(final_model)$coefficients),
          "/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/Report4/final_model_output.csv")

plot(final_model$model[,1],residuals(final_model))
View(training_data[which(residuals(final_model) < -2),])

validation_data$Predicted_price_log <- predict(final_model, newdata = validation_data)
validation_data$Predicted_price <- exp(validation_data$Predicted_price_log)
observed_values <- validation_data$price
predicted_values <- validation_data$Predicted_price
data.frame(rmse = RMSE(predicted_values, observed_values),
            mae = MAE(predicted_values, observed_values),
            mape = MAPE(predicted_values, observed_values),
            r_squared = R2_Score(predicted_values, observed_values))
training_data$Predicted_price_log <- predict(final_model, newdata = training_data)
training_data$Predicted_price <- exp(training_data$Predicted_price_log)
fig4_data <- rbind(training_data,validation_data)

ggplot(data = fig4_data) + aes(x = price_log, y = Predicted_price_log, color = obs_type) +
        geom_point() +
        geom_abline(intercept = 0, slope = 1, linetype="dashed") +
        theme_bw() +
        theme(legend.position = "bottom",
        legend.title = element_blank()) +
        labs(x = "Acvtual Value", y = "Predicted Value")
## fit the full model with all predictors
m.mlr <- lm(formula = price_log ~ car_age + engine_power + car_type + feature_sum +
              mileage + model_key + paint_color + fuel +
              feature_1 + feature_2 + feature_3 + feature_4 +
              feature_5 + feature_6 + feature_8,
            data = training_data)
summary(m.mlr)

set.seed(121023)
## LASSO ----
## define lasso model
lasso_model <- glmnet(x = model.matrix(m.mlr)[,-1],
                      y = training_data$price_log,
                      alpha = 1)
## plot the coef path vs lambda
plot(lasso_model, xvar = "lambda", label = TRUE)
## cross-validation
cv_model <- cv.glmnet(x = model.matrix(m.mlr)[,-1],
                      y = training_data$price_log,
                      alpha = 1, nfolds = 10)
## extract the best lambda value
best_lasso_lambda <- cv_model$lambda.min
cat("Best Lambda - LASSO:", best_lasso_lambda)
```

```r
## extract coefficients from the lasso model at the best lambda value
lasso_coef <- coef(lasso_model, s = best_lasso_lambda)
print(lasso_coef)
## make prediction using the best lasso model in the validate set
lasso_pred_price_log <- predict(lasso_model, s = best_lasso_lambda, newx = val.mat[,-1])
lasso_pred_price <- exp(lasso_pred_price_log)

## Ridge Regression ----
## define ridge regression model
ridge_model <- glmnet(x = model.matrix(m.mlr)[,-1],
                      y = training_data$price_log,
                      alpha = 0)
## plot the coef path vs lambda
plot(ridge_model, xvar = "lambda", label = TRUE)
## cross-validation
cv_model <- cv.glmnet(x = model.matrix(m.mlr)[,-1],
                      y = training_data$price_log,
                      alpha = 0, nfolds = 10)
## extract the best lambda value
best_ridge_lambda <- cv_model$lambda.min
cat("Best Lambda - Ridge:", best_ridge_lambda)
## extract coefficients from the ridge regression at the best lambda value
ridge_coef <- coef(ridge_model, s = best_ridge_lambda)
print(ridge_coef)
## make prediction using the best ridge regression in the validate set
ridge_pred_price_log <- predict(ridge_model, s = best_ridge_lambda, newx = val.mat[,-1])
ridge_pred_price <- exp(ridge_pred_price_log)
obs_price <- validation_data$price
prediction_metrics <- rbind(data.frame(model     = "LASSO",
                                       rmse      = RMSE(lasso_pred_price, obs_price),
                                       mae       = MAE(lasso_pred_price, obs_price),
                                       mape      = MAPE(lasso_pred_price, obs_price),
                                       r_squared = R2_Score(lasso_pred_price, obs_price)),
                            data.frame(model     = "Ridge",
                                       rmse      = RMSE(ridge_pred_price, obs_price),
                                       mae       = MAE(ridge_pred_price, obs_price),
                                       mape      = MAPE(ridge_pred_price, obs_price),
                                       r_squared = R2_Score(ridge_pred_price, obs_price))
                            )
prediction_metrics
#   model     rmse      mae      mape r_squared
# 1 LASSO 3589.902 2345.031 0.5394894 0.8197743
# 2 Ridge 3611.291 2345.482 0.5412341 0.8176203
coef <- data.frame(LASSO=as.matrix(lasso_coef),
                   ridge=as.matrix(ridge_coef))
write.csv(coef,"/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/FinalReport/lasso_ridge_coef.csv")
## figure of coefs vs lambda
pdf("/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/FinalReport/Figure4.pdf",
    height = 4,width = 8)
par(mfrow=c(1,2))
plot(lasso_model, xvar = "lambda", label = TRUE)
plot(ridge_model, xvar = "lambda", label = TRUE)
dev.off()
```

```
## figure of actual value vs fitted value
final_ols_model <- lm(formula = price_log ~ car_age + engine_power +
                        car_type + feature_sum +
                    mileage + model_key + paint_color + fuel + feature_3 +
                      feature_6 + feature_5 + feature_4, data = training_data)

bmw2.mat <- model.matrix(lm(formula = rep(1,dim(bmw2)[1]) ~ car_age + engine_power + car_type + feature_
                             mileage + model_key + paint_color + fuel +
                             feature_1 + feature_2 + feature_3 + feature_4 +
                             feature_5 + feature_6 + feature_8,
                           data = bmw2))

bmw2$ols_pred_price_log <- predict(final_ols_model, newdata = bmw2)
bmw2$lasso_pred_price_log <- predict(lasso_model, s = best_lasso_lambda, newx = bmw2.mat[,-1])
bmw2$ridge_pred_price_log <- predict(ridge_model, s = best_ridge_lambda, newx = bmw2.mat[,-1])

bmw2.pred <- data.frame(y=rep(bmw2$price_log,3),
                        obs_type=rep(bmw2$obs_type,3),
                      yhat=c(bmw2$ols_pred_price_log,
                             bmw2$lasso_pred_price_log,
                             bmw2$ridge_pred_price_log),
                        model=rep(c("OLS","LASSO","Ridge"),each = dim(bmw2)[1])
                        )
bmw2.pred$model <- factor(bmw2.pred$model,levels = c("OLS","LASSO","Ridge"))
library(ggplot2)
pdf("/Users/kurt/Desktop/BU_Fall_2023/MA575/Project/FinalReport/Figure5.pdf",
    height = 4,width = 10)
ggplot(data = bmw2.pred) + aes(x = y, y = yhat, color = obs_type) +
  geom_point(alpha=0.6) +
  geom_abline(intercept = 0, slope = 1, linetype="dashed") +
  xlim(4.5,12.1) + ylim(4.5,12.1) +
  facet_wrap(~model) +
  theme_bw() +
  # theme(legend.title = element_blank(), legend.position = "bottom") +
  theme(legend.title = element_blank(),
        # # Remove panel border
        # panel.border = element_blank(),
        # Remove panel grid lines
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        # Remove panel background
        panel.background = element_blank(),
        legend.position = c(0.94, 0.12), # c(0,0) bottom left, c(1,1) top-right.
        legend.background = element_rect(fill = "white", colour = "NA")) +
  xlab("Observed Value") + ylab("Predicted Value")
dev.off()
# Investigate the cars with low price -------------------------------------
bmw2%>%
  mutate(low_price=ifelse(price<1000,T,F))%>%
  group_by(low_price)%>%
  summarise(meanMileage=mean(mileage),
            sdMileage=sd(mileage),
            meanEnginePower=mean(engine_power),
            sdEnginePower=sd(engine_power),
            meanCarAge=mean(car_age),
            sdCarAge=sd(car_age),
            meanNFeatures=mean(feature_sum),
            sdNFeatures=sd(feature_sum))
```