Building an Used Car Price Prediction Model for Germany eBay listings

Richa Gupta/Sandro Tanis/Ping Wang

Aug 07, 2020

Table of Contents

# 1. Introduction

# 2. Methods

## 2.1 Data cleanning and exploration

Loading for raw data

library(readr)  
autos\_raw <- read\_csv("autos.csv")  
attr(autos\_raw, 'spec') <- NULL  
attr(autos\_raw, 'problems') <- NULL  
  
str(autos\_raw)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 354687 obs. of 20 variables:  
## $ dateCrawled : POSIXct, format: "2016-03-24 11:52:17" "2016-03-24 10:58:45" ...  
## $ name : chr "Golf\_3\_1.6" "A5\_Sportback\_2.7\_Tdi" "Jeep\_Grand\_Cherokee\_\"Overland\"" "GOLF\_4\_1\_4\_\_3T\xdcRER" ...  
## $ seller : chr "privat" "privat" "privat" "privat" ...  
## $ offerType : chr "Angebot" "Angebot" "Angebot" "Angebot" ...  
## $ price : num 480 18300 9800 1500 3600 650 2200 0 14500 999 ...  
## $ abtest : chr "test" "test" "test" "test" ...  
## $ vehicleType : chr NA "coupe" "suv" "kleinwagen" ...  
## $ yearOfRegistration : num 1993 2011 2004 2001 2008 ...  
## $ gearbox : chr "manuell" "manuell" "automatik" "manuell" ...  
## $ powerPS : num 0 190 163 75 69 102 109 50 125 101 ...  
## $ model : chr "golf" NA "grand" "golf" ...  
## $ kilometer : num 150000 125000 125000 150000 90000 150000 150000 40000 30000 150000 ...  
## $ monthOfRegistration: num 0 5 8 6 7 10 8 7 8 0 ...  
## $ fuelType : chr "benzin" "diesel" "diesel" "benzin" ...  
## $ brand : chr "volkswagen" "audi" "jeep" "volkswagen" ...  
## $ notRepairedDamage : chr NA "ja" NA "nein" ...  
## $ dateCreated : POSIXct, format: "2016-03-24" "2016-03-24" ...  
## $ nrOfPictures : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ postalCode : chr "70435" "66954" "90480" "91074" ...  
## $ lastSeen : POSIXct, format: "2016-04-07 03:16:57" "2016-04-07 01:46:50" ...

## 2.1.1 Data cleaning and preparation

autos=subset(autos\_raw, price>0 & yearOfRegistration>=2000 & offerType=="Angebot" & seller=="privat")  
columns\_remove=c("postalCode","lastSeen", "nrOfPictures","dateCreated","dateCrawled","monthOfRegistration","offerType","seller")   
columns\_numeric = c("price","powerPS", "kilometer","yearOfRegistration")  
columns\_factor = c("abtest","vehicleType","gearbox","model","brand","fuelType","notRepairedDamage" )  
other\_columns = c("name")   
  
#autos[,columns\_factor]=lapply(autos[,columns\_factor], as.factor)  
autos = autos[, -which(names(autos) %in% columns\_remove)]  
autos = autos[, -which(names(autos) %in% other\_columns)]  
str(autos)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 250368 obs. of 11 variables:  
## $ price : num 18300 9800 1500 3600 2200 ...  
## $ abtest : chr "test" "test" "test" "test" ...  
## $ vehicleType : chr "coupe" "suv" "kleinwagen" "kleinwagen" ...  
## $ yearOfRegistration: num 2011 2004 2001 2008 2004 ...  
## $ gearbox : chr "manuell" "automatik" "manuell" "manuell" ...  
## $ powerPS : num 190 163 75 69 109 125 105 140 131 190 ...  
## $ model : chr NA "grand" "golf" "fabia" ...  
## $ kilometer : num 125000 125000 150000 90000 150000 30000 150000 150000 150000 70000 ...  
## $ fuelType : chr "diesel" "diesel" "benzin" "diesel" ...  
## $ brand : chr "audi" "jeep" "volkswagen" "skoda" ...  
## $ notRepairedDamage : chr "ja" NA "nein" "nein" ...

## 2.1.2 Data exploration to find the relationship between continous variables

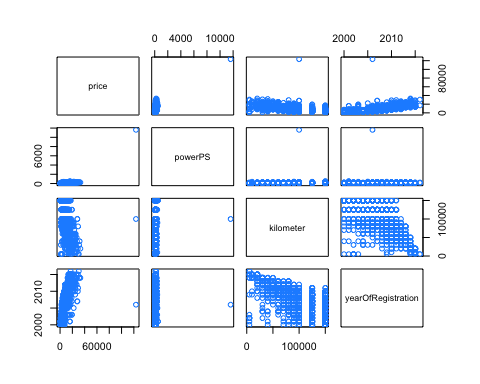
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

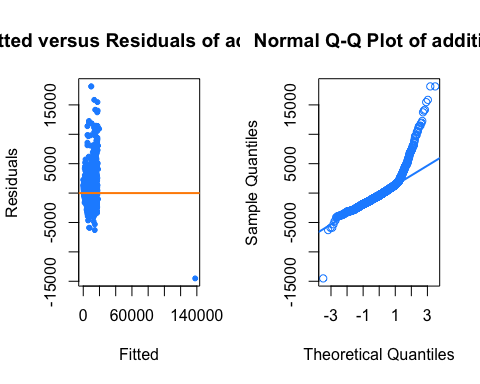
autos=na.omit(autos)  
autos\_factor\_groups=autos %>% count (abtest,vehicleType,gearbox,model,brand,fuelType,notRepairedDamage)  
autos\_factor\_groups=autos\_factor\_groups[order(autos\_factor\_groups$n,decreasing = TRUE),]  
group1=autos\_factor\_groups[2,]  
group1.size = group1$n  
group1=subset(group1, select = -c(n) )  
  
autos\_1=autos  
cols=names(group1)  
for (i in 1:ncol(group1)){  
 idx = autos\_1[,cols[i]]==group1[[i]]  
 autos\_1=autos\_1[idx,]  
}  
  
autos\_1=subset(autos\_1, select = columns\_numeric )  
   
  
#  
pairs(autos\_1,col="dodgerblue")



### Autos\_1 has been isolated to better find a model format for continuous variables

Please use **autos\_1** for determining box-cox tranformations,it only contains continuous variables

par(mfrow=c(1,2))  
model\_add = lm(price ~ powerPS + kilometer+yearOfRegistration , data = autos\_1)  
#fitted vs residual  
plot(fitted(model\_add), resid(model\_add), col = "dodgerblue", pch = 20,  
 xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals of additive")  
abline(h = 0, col = "darkorange", lwd = 2)  
  
#qqplot  
qqnorm(resid(model\_add), main = "Normal Q-Q Plot of additive", col = "dodgerblue")  
qqline(resid(model\_add), col = "dodgerblue", lwd = 2)



## 2.1.3 perform box-cox tranformation on continuous variables only (to estabish transformation form )

library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

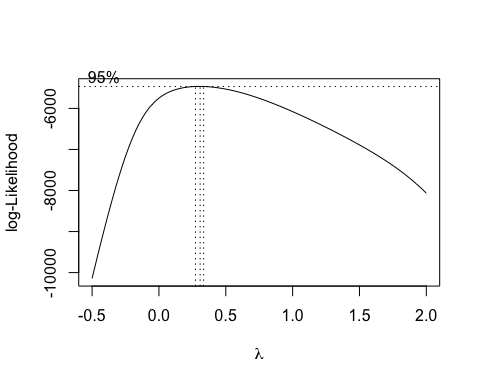
library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

par(mfrow=c(1,1))  
out=boxcox(model\_add, plotit = TRUE, lambda = seq(-0.5, 2.0, by = 0.1))



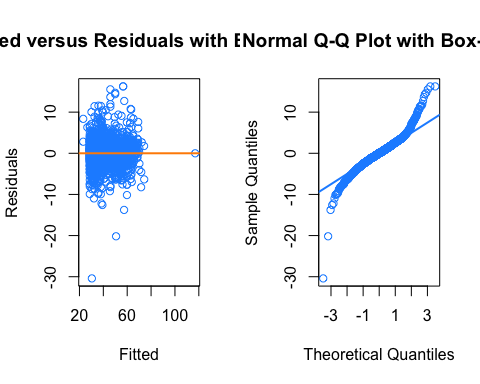
( lambda=out$x[which.max(out$y)] )

## [1] 0.3081

model\_cox\_start=lm( (price^lambda-1)/lambda ~ powerPS + kilometer+yearOfRegistration+I(powerPS^2) + I(kilometer^2) + I(log(kilometer)), data = autos\_1)  
model\_cox = step(model\_cox\_start,trace=0)  
#model\_cox  
#model\_cox2 = step(model\_cox\_start,k=log(nrow(autos\_1)),trace=0)  
#model\_cox2  
  
#model\_add\_cox=lm((price^lambda-1)/lambda ~ powerPS, data = autos\_1)  
  
summary(model\_cox)

##   
## Call:  
## lm(formula = (price^lambda - 1)/lambda ~ powerPS + yearOfRegistration +   
## I(powerPS^2) + I(kilometer^2) + I(log(kilometer)), data = autos\_1)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -30.408 -1.712 0.036 1.653 16.253   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -3.43e+03 5.32e+01 -64.52 <2e-16 \*\*\*  
## powerPS 7.42e-02 1.38e-03 53.65 <2e-16 \*\*\*  
## yearOfRegistration 1.73e+00 2.63e-02 65.68 <2e-16 \*\*\*  
## I(powerPS^2) -5.79e-06 1.20e-07 -48.13 <2e-16 \*\*\*  
## I(kilometer^2) -2.67e-10 1.80e-11 -14.83 <2e-16 \*\*\*  
## I(log(kilometer)) 3.99e-01 2.25e-01 1.78 0.076 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.18 on 2023 degrees of freedom  
## Multiple R-squared: 0.918, Adjusted R-squared: 0.918   
## F-statistic: 4.52e+03 on 5 and 2023 DF, p-value: <2e-16

par(mfrow=c(1,2))  
plot(fitted(model\_cox), resid(model\_cox), col = "dodgerblue",   
 xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals with Box-cox")  
abline(h = 0, col = "darkorange", lwd = 2)  
qqnorm(resid(model\_cox), main = "Normal Q-Q Plot with Box-cox", col = "dodgerblue")  
qqline(resid(model\_cox), col = "dodgerblue", lwd = 2)



shapiro.test(resid(model\_cox))

##   
## Shapiro-Wilk normality test  
##   
## data: resid(model\_cox)  
## W = 0.94, p-value <2e-16

bptest(model\_cox)

##   
## studentized Breusch-Pagan test  
##   
## data: model\_cox  
## BP = 78, df = 5, p-value = 2e-15

### tempory block - refit the multiple regression model without any influential points

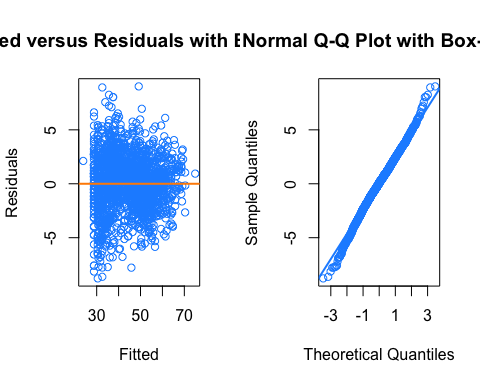
#finding influenctial  
mod\_cook = cooks.distance(model\_cox)  
n=length(resid(model\_cox))  
high\_infl = mod\_cook > 4 / n  
sum(high\_infl)

## [1] 96

mean(high\_infl)

## [1] 0.04731

#Refit the multiple regression model without any influential points  
formula=as.formula(as.character(model\_cox$call[2]))  
model\_cox\_sub = lm(formula, data = autos\_1, subset = !high\_infl)  
par(mfrow=c(1,2))  
plot(fitted(model\_cox\_sub), resid(model\_cox\_sub), col = "dodgerblue",   
 xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals with Box-cox")  
abline(h = 0, col = "darkorange", lwd = 2)  
qqnorm(resid(model\_cox\_sub), main = "Normal Q-Q Plot with Box-cox", col = "dodgerblue")  
qqline(resid(model\_cox\_sub), col = "dodgerblue", lwd = 2)



summary(model\_cox\_sub)

##   
## Call:  
## lm(formula = formula, data = autos\_1, subset = !high\_infl)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -8.774 -1.536 0.074 1.623 9.027   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -3.39e+03 4.60e+01 -73.85 <2e-16 \*\*\*  
## powerPS 6.31e-02 4.25e-03 14.85 <2e-16 \*\*\*  
## yearOfRegistration 1.71e+00 2.26e-02 75.54 <2e-16 \*\*\*  
## I(powerPS^2) 3.47e-05 1.24e-05 2.79 0.0053 \*\*   
## I(kilometer^2) -2.57e-10 1.51e-11 -17.09 <2e-16 \*\*\*  
## I(log(kilometer)) 1.92e-01 2.08e-01 0.92 0.3555   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.5 on 1927 degrees of freedom  
## Multiple R-squared: 0.945, Adjusted R-squared: 0.945   
## F-statistic: 6.6e+03 on 5 and 1927 DF, p-value: <2e-16

#(coef(int\_model) - coef(int\_model\_sub)) / coef(int\_model)  
shapiro.test(resid(model\_cox\_sub))

##   
## Shapiro-Wilk normality test  
##   
## data: resid(model\_cox\_sub)  
## W = 1, p-value = 0.0004

bptest(model\_cox\_sub)

##   
## studentized Breusch-Pagan test  
##   
## data: model\_cox\_sub  
## BP = 133, df = 5, p-value <2e-16

## 2.1.4 determine best lambda for box-cox tranformation

We will run box-cox transformation on subgroup of data that has more than 300 records..

source("misc\_functions.R")  
(boxcox\_lambda=subset\_autodata\_with\_boxcox(autos))  
hist(boxcox\_lambda,breaks=20,col="lightblue")  
mean(boxcox\_lambda)

Based on our analysis above, we will use for the Box-Cox transformation!

lambda=0.3  
formula\_str=as.character(model\_cox$call[2])

### Phase II - adding categorical variable to the model form determine in Phase I

##### The best model form with only continous variable is:

(price^lambda - 1)/lambda ~ powerPS + yearOfRegistration + I(powerPS^2) + I(kilometer^2) + I(log(kilometer))

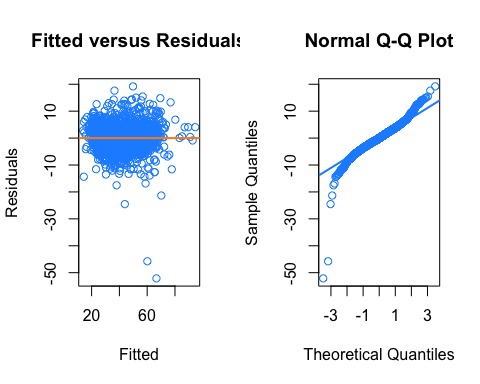
#### Next, we combine continous variables(wiht established form) with all factor varible to start a backward AIC to find a good model.

size\_train=2000  
  
idx\_train=sample(1:nrow(autos),size\_train)  
autos\_train=autos[idx\_train,]  
autos\_train[,columns\_factor]=lapply(autos\_train[,columns\_factor], as.factor)  
#model2\_start = lm( ((price^lambda - 1)/lambda) ~ .^2 + I(powerPS^2) + I(kilometer^2) + I(log(kilometer)), autos\_train )   
model2\_start = lm( ((price^lambda - 1)/lambda) ~ . + I(powerPS^2) + I(kilometer^2) + I(log(kilometer)), data=autos\_train )   
n=nrow(autos\_train)  
model2\_selected\_bic = step(model2\_start,k=log(n),trace=0)  
model2\_selected\_bic

##   
## Call:  
## lm(formula = ((price^lambda - 1)/lambda) ~ vehicleType + yearOfRegistration +   
## gearbox + powerPS + fuelType + brand + notRepairedDamage +   
## I(powerPS^2) + I(kilometer^2), data = autos\_train)  
##   
## Coefficients:  
## (Intercept) vehicleTypebus vehicleTypecabrio   
## -3.48e+03 1.31e+00 3.96e+00   
## vehicleTypecoupe vehicleTypekleinwagen vehicleTypekombi   
## 1.89e+00 -2.16e+00 -1.69e+00   
## vehicleTypelimousine vehicleTypesuv yearOfRegistration   
## -3.69e-01 3.85e+00 1.75e+00   
## gearboxmanuell powerPS fuelTypecng   
## -1.46e+00 4.56e-02 -8.39e-01   
## fuelTypediesel fuelTypehybrid fuelTypelpg   
## 1.83e+00 3.31e+00 2.78e-01   
## brandaudi brandbmw brandchevrolet   
## 6.19e+00 5.69e+00 -6.24e+00   
## brandchrysler brandcitroen branddacia   
## -2.86e-02 -1.88e+00 -4.16e+00   
## branddaewoo branddaihatsu brandfiat   
## -2.99e+00 -8.25e-01 -1.70e+00   
## brandford brandhonda brandhyundai   
## -2.61e-02 3.11e+00 -2.04e+00   
## brandjaguar brandjeep brandkia   
## 3.11e+00 9.16e+00 -1.98e+00   
## brandlancia brandland\_rover brandmazda   
## -1.59e+00 3.06e+00 1.48e-01   
## brandmercedes\_benz brandmini brandmitsubishi   
## 5.84e+00 7.07e+00 1.91e+00   
## brandnissan brandopel brandpeugeot   
## -5.69e-01 3.33e-01 -7.04e-01   
## brandporsche brandrenault brandrover   
## 2.02e+01 -1.69e+00 -1.02e+01   
## brandsaab brandseat brandskoda   
## -4.56e-02 1.42e+00 2.21e+00   
## brandsmart brandsubaru brandsuzuki   
## -4.31e-01 2.35e+00 -3.96e-01   
## brandtoyota brandvolkswagen brandvolvo   
## 3.08e+00 4.31e+00 3.38e+00   
## notRepairedDamagenein I(powerPS^2) I(kilometer^2)   
## 5.72e+00 3.49e-05 -2.42e-10

## 2.1.4 remove high influence data and refit the choose model

model =model2\_selected\_bic  
par(mfrow=c(1,2))  
plot(fitted(model), resid(model), col = "dodgerblue",   
 xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")  
abline(h = 0, col = "darkorange", lwd = 2)  
qqnorm(resid(model), main = "Normal Q-Q Plot", col = "dodgerblue")  
qqline(resid(model), col = "dodgerblue", lwd = 2)



shapiro.test(resid(model))

##   
## Shapiro-Wilk normality test  
##   
## data: resid(model)  
## W = 0.92, p-value <2e-16

bptest(model)

##   
## studentized Breusch-Pagan test  
##   
## data: model  
## BP = 75, df = 53, p-value = 0.02

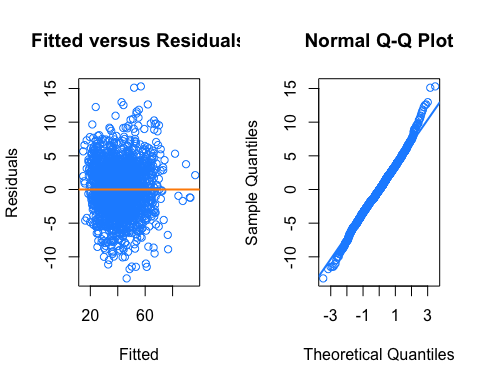
#finding influenctial  
mod\_cook = cooks.distance(model)  
n=length(resid(model))  
high\_infl = mod\_cook > 4 / n  
sum(high\_infl)

## [1] NA

mean(high\_infl)

## [1] NA

#Refit the multiple regression model without any influential points  
formula=as.formula(as.character(model$call[2]))  
model\_sub = lm(formula, data = autos\_train, subset = !high\_infl)  
par(mfrow=c(1,2))  
plot(fitted(model\_sub), resid(model\_sub), col = "dodgerblue",   
 xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals")  
abline(h = 0, col = "darkorange", lwd = 2)  
qqnorm(resid(model\_sub), main = "Normal Q-Q Plot", col = "dodgerblue")  
qqline(resid(model\_sub), col = "dodgerblue", lwd = 2)



shapiro.test(resid(model\_sub))

##   
## Shapiro-Wilk normality test  
##   
## data: resid(model\_sub)  
## W = 1, p-value = 2e-05

bptest(model\_sub)

##   
## studentized Breusch-Pagan test  
##   
## data: model\_sub  
## BP = 151, df = 49, p-value = 3e-12

summary(model\_sub)$adj.r.squared

## [1] 0.9109

coef(summary(model\_sub))

## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -3.554e+03 6.308e+01 -56.33347 0.000e+00  
## vehicleTypebus 6.837e-01 2.187e+00 0.31270 7.545e-01  
## vehicleTypecabrio 2.951e+00 2.206e+00 1.33724 1.813e-01  
## vehicleTypecoupe 8.403e-01 2.223e+00 0.37800 7.055e-01  
## vehicleTypekleinwagen -2.798e+00 2.182e+00 -1.28241 1.999e-01  
## vehicleTypekombi -2.122e+00 2.181e+00 -0.97289 3.307e-01  
## vehicleTypelimousine -9.863e-01 2.181e+00 -0.45215 6.512e-01  
## vehicleTypesuv 3.621e+00 2.218e+00 1.63243 1.028e-01  
## yearOfRegistration 1.787e+00 3.146e-02 56.80551 0.000e+00  
## gearboxmanuell -1.214e+00 2.404e-01 -5.05035 4.844e-07  
## powerPS 4.004e-02 4.591e-03 8.72008 6.066e-18  
## fuelTypecng -3.989e-01 1.545e+00 -0.25820 7.963e-01  
## fuelTypediesel 1.474e+00 2.224e-01 6.62953 4.405e-11  
## fuelTypelpg -2.600e-01 7.084e-01 -0.36703 7.136e-01  
## brandaudi 7.068e+00 1.356e+00 5.21286 2.067e-07  
## brandbmw 7.061e+00 1.352e+00 5.22456 1.943e-07  
## brandchevrolet -5.395e+00 1.721e+00 -3.13485 1.746e-03  
## brandchrysler 2.437e+00 1.943e+00 1.25391 2.100e-01  
## brandcitroen -1.118e+00 1.460e+00 -0.76597 4.438e-01  
## branddacia -4.386e+00 1.796e+00 -2.44264 1.467e-02  
## branddaewoo -2.386e+00 3.972e+00 -0.60069 5.481e-01  
## branddaihatsu 6.182e-01 2.544e+00 0.24303 8.080e-01  
## brandfiat -5.252e-01 1.462e+00 -0.35915 7.195e-01  
## brandford 6.270e-01 1.364e+00 0.45958 6.459e-01  
## brandhonda 3.736e+00 1.712e+00 2.18229 2.921e-02  
## brandhyundai -1.080e+00 1.558e+00 -0.69318 4.883e-01  
## brandjaguar 3.548e+00 3.971e+00 0.89349 3.717e-01  
## brandkia -1.356e+00 1.611e+00 -0.84168 4.001e-01  
## brandland\_rover 4.476e+00 2.066e+00 2.16681 3.038e-02  
## brandmazda 1.726e+00 1.488e+00 1.15983 2.463e-01  
## brandmercedes\_benz 6.782e+00 1.356e+00 5.00180 6.217e-07  
## brandmini 7.748e+00 1.500e+00 5.16620 2.647e-07  
## brandmitsubishi 2.389e+00 2.533e+00 0.94329 3.457e-01  
## brandnissan -6.817e-01 1.488e+00 -0.45798 6.470e-01  
## brandopel 1.145e+00 1.356e+00 0.84429 3.986e-01  
## brandpeugeot 9.001e-02 1.416e+00 0.06358 9.493e-01  
## brandporsche 2.084e+01 1.882e+00 11.07155 1.244e-27  
## brandrenault -1.077e+00 1.386e+00 -0.77735 4.370e-01  
## brandsaab 1.991e+00 2.958e+00 0.67322 5.009e-01  
## brandseat 1.711e+00 1.449e+00 1.18054 2.379e-01  
## brandskoda 2.373e+00 1.440e+00 1.64715 9.970e-02  
## brandsmart -4.205e-01 1.562e+00 -0.26918 7.878e-01  
## brandsubaru 4.521e+00 2.559e+00 1.76684 7.742e-02  
## brandsuzuki 1.457e-01 1.605e+00 0.09078 9.277e-01  
## brandtoyota 3.597e+00 1.563e+00 2.30188 2.145e-02  
## brandvolkswagen 5.068e+00 1.341e+00 3.77890 1.625e-04  
## brandvolvo 4.499e+00 1.685e+00 2.66994 7.653e-03  
## notRepairedDamagenein 5.259e+00 3.276e-01 16.05422 2.112e-54  
## I(powerPS^2) 5.466e-05 1.149e-05 4.75781 2.109e-06  
## I(kilometer^2) -2.417e-10 1.527e-11 -15.82214 5.530e-53

calc\_loocv\_rmse = function(model) {  
 sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))  
}

## test results

TEST result?

LOOC\_RMSE CV

# 3. Results

# 4. Discussion

# 5. Appendix