Building an Used Car Price Prediction Model for Germany eBay listings

Richa Gupta/Sandro Tanis/Ping Wang

Aug 07, 2020

---

1. Introduction

The formation of our group hinged upon our interest in the automotive industry, as a result of that we have chosen this robust dataset because we wanted to have a better understanding about the used car market and this dataset had all of the components that we wanted to observe for this project.The dataset that I am using in this project was found on Kaggle, the well-known Machine Learning Competition website and it is about Used Car listings from eBay – Germany.

The dataset we are using for this project was found on Kaggle, the well-known Machine Learning Competition website and it is about Used Car listings from eBay – Germany please see link here: https://www.kaggle.com/orgesleka/used-cars-database/home (https://www.kaggle.com/orgesleka/used-cars-database/home).

- Data file contains approximately **370,000 observations** and **20 variables** that are scraped from used-car listing on Ebay-Kleinanzeigen (German). The high quantity and authenticity makes this dataset useful for an exploratory analysis.

- Below is the list of various variables associated with the dataset:
    - **name**: Name of the car
    - **abtest**: Indicates the A/B testing group used by ebay website team
    - **seller**: Whether the seller is private or a dealer
    - **offerType**: The type of listing
    - **price**: The price on the ad to sell the car.
    - **vehicleType**: The vehicle Type.
    - **yearOfRegistration**: The year in which year the car was first registered.
    - **gearbox**: The transmission type.
    - **powerPS**: The power of the car in PS.
    - **model**: The car model name.
    - **kilometer**: How many kilometers the car has driven.
    - **monthOfRegistration**: The month in which year the car was first registered.
    - **fuelType**: What type of fuel the car uses.
    - **brand**: The brand of the car.
    - **notRepairedDamage**: If the car has a damage which is not yet repaired.
    - **postalCode**: The postal code for the location of the vehicle.
- We considered cars that were registered after the year 2000, so we can build a good model based on the last 20 years in the used auto industry in Germany.
- We randomly sampled 100,000 records after our data cleaning process, and split half the data into our training dataset and test dataset which resulted in 50,000 records each for our model prediction. We used below variables for our final analysis.
    - **price**: The price on the ad to sell the car.
    - **abtest**: Indicates the A/B testing group used by ebay website team
    - **vehicleType**: The vehicle Type.
    - **yearOfRegistration**: The year in which year the car was first registered.
    - **gearbox**: The transmission type.
    - **powerPS**: The power of the car in PS.
    - **kilometer**: How many kilometers the car has driven.
    - **fuelType**: What type of fuel the car uses.
    - **notRepairedDamage**: If the car has a damage which is not yet repaired.

This project focuses on finding a good model for predicting based on the statistical knowledge that we gained from the class. We tried to determine the relationship between different variables, between the chosen predictor variable and price. Furthermore, our goal is to build a MLR model for used car price prediction where we use price as the response variable, the predictors will be selected from all remaining continuous numerical variables such as (PowerPs, Kilometer, yearRegistrations) and the other categorical variables so we can make compare 2 different methods so we can arrive to the best model that will fit our prediction. We will also be using the leftover observations to validate our model performance trough examining our assumptions by using regression diagnostic testing. Other method that we will be using are collinearity to check for issues on model and dummy

variables for categorical fields. Finally, additional methods such as: outlier diagnostics, polynomial regression, and transformations will be leveraged when needed to further analyze and prove that our final model is justified.

2. Methods

In this section, we will start with data cleaning steps to remove observations that seem unreasonable. We will then follow by steps to remove unwanted columns and creating datasets for training(model building) and testing. We will create one training dataset consists of 50,000 randomly sampled record, and likewise for the testing dataset. We will also examine continuous numeric predictors for any collinearity issues.

With the training dataset, we will take methods to find a set of good candidate models.

2.0 Data preparation

2.0.1 Loading necessary Libaries and the script file

The major libraries needed will be loaded from here to be able to be used throughout the document. This will include the libraries such as LMTEST to test linear regression model, read input from a csv files readr, and dplyr which enables us to manipulate the dataset.

```
library(dplyr)
library(readr)
library(lmtest)
library(MASS)
source("misc_functions.R")
```

2.0.2 Data loading

Loading the raw data from the CSV file:

```
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 vari
ables:
##  $ dateCrawled        : POSIXct, format: "2016-03-24 11:52:17" "2016-03-24 10:5
8:45" ...
##  $ name               : chr  "Golf_3_1.6" "A5_Sportback_2.7_Tdi" "Jeep_Grand_Che
rokee_\"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 400
00 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 01:4
6:50" ...
```

2.0.3 Data cleaning and unused columns removal

```
autos=subset(autos_raw, price>500 & price<200000 & yearOfRegistration>=2000 & power
PS>0 & offerType=="Angebot" & seller=="privat")
columns_remove=c("postalCode","lastSeen","name","model","brand", "nrOfPictures","da
teCreated","dateCrawled","monthOfRegistration","offerType","seller")
columns_numeric = c("price","powerPS","yearOfRegistration","kilometer")
columns_factor = c("abtest","vehicleType","gearbox","fuelType","notRepairedDamage"
 )
autos=na.omit(autos)
autos = autos[, -which(names(autos) %in% columns_remove)]
```

Our data cleaning process involved removing not-so-useful variables such as postalCode, lastSeen, name etc.. and clean the data by removing NAs and other values that might be not of significance.

After loading the raw data, **autos_raw** we:

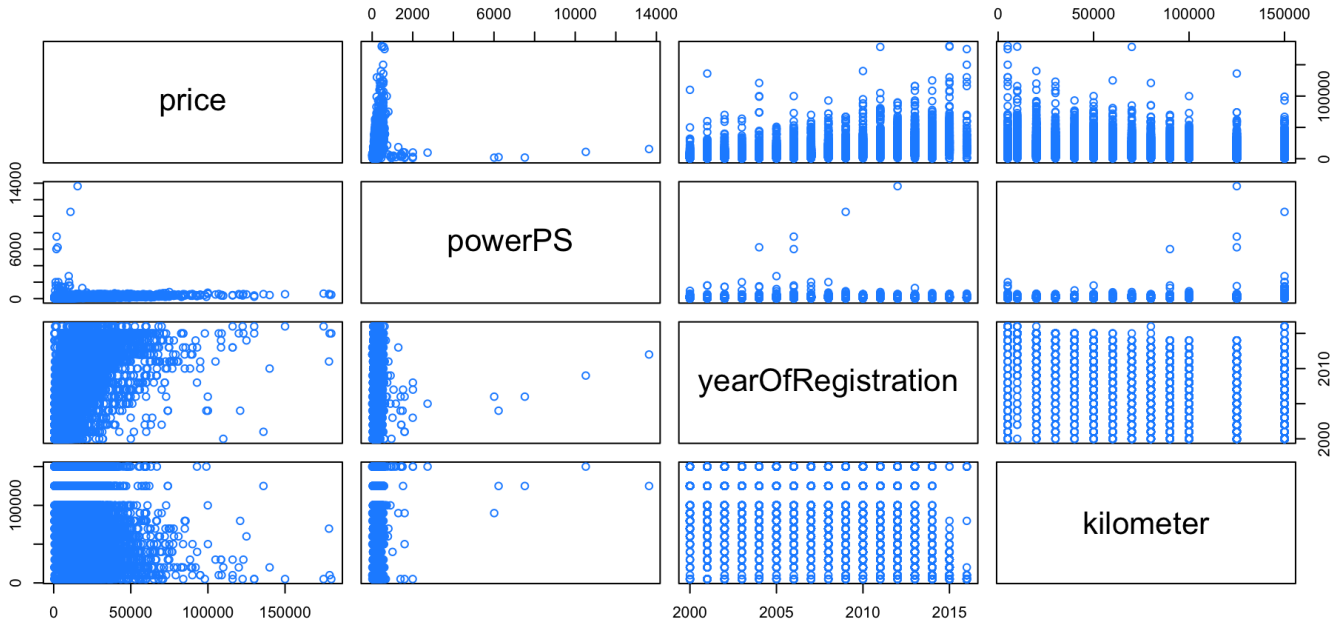
- Removed data with *price* < 0 (because a car cannot be associated with $0 pricing)

- Keep only data with offerType="Angebot" and seller=="privat", (with an exception of a few observations all data is associated with these values)
- Keep only data with YearOfRegistration>=2000, (to focus only on last 20 years worth of registered cars)
- Removed unused columns: postalCode, lastSeen, name, model, brand, nrOfPictures, dateCreated, dateCrawled, monthOfRegistration, offerType, seller, (for better analysis and predictions)
- Identify continous numeric columns : price, powerPS, yearOfRegistration, kilometer, (to help with our methods)
- Identify factor columns : abtest, vehicleType, gearbox, fuelType, notRepairedDamage, (to help with our methods)

2.0.4 Checking collinearity on numberic variable

Taking a look at the continuous variables only - checking for collinearity

```
#Checking colinear on numberic columns on 50,000 records sample
sample_size=50000
idx_sample=sample(1:nrow(autos),sample_size)
autos_sample= subset (autos[idx_sample,], select = columns_numeric)
pairs(autos_sample,col="dodgerblue")
```




```
round(cor(autos_sample), 2)
```

```
##                     price powerPS yearOfRegistration kilometer
## price                1.00    0.34               0.61     -0.46
## powerPS              0.34    1.00               0.09      0.00
## yearOfRegistration   0.61    0.09               1.00     -0.62
## kilometer           -0.46    0.00              -0.62      1.00
```

Based on the above Pair plot and the output of cor() function, we have determined there no significant collinearity issues. Therefore, we can proceed to include all continuous variables (except for Price) as model predictors.

2.0.5 Setup training and testing data

```
#Setup data with 100,000 randomly sampled and all columns
training_size=50000
set.seed(20200807)
idx_train=sample(1:nrow(autos),training_size)
autos_train= autos[idx_train,]
autos_train[,columns_factor]=lapply(autos_train[,columns_factor], as.factor)
str(autos_train)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    50000 obs. of  9 variables:
##  $ price             : num  1600 11950 4399 10900 3999 ...
##  $ abtest            : Factor w/ 2 levels "control","test": 2 2 2 1 1 1 1 1 2 1
...
##  $ vehicleType       : Factor w/ 8 levels "andere","bus",..: 2 7 7 7 7 5 6 6 5 6
...
##  $ yearOfRegistration: num  2004 2009 2003 2009 2008 ...
##  $ gearbox           : Factor w/ 2 levels "automatik","manuell": 2 1 2 2 2 2 2 2
2 1 ...
##  $ powerPS           : num  75 177 140 170 75 101 170 270 103 184 ...
##  $ kilometer         : num  150000 150000 150000 150000 70000 150000 150000 1500
00 90000 30000 ...
##  $ fuelType          : Factor w/ 7 levels "andere","benzin",..: 4 4 4 4 2 4 4 4
2 4 ...
##  $ notRepairedDamage : Factor w/ 2 levels "ja","nein": 2 2 2 2 2 2 2 2 2 2 ...
##  - attr(*, "na.action")= 'omit' Named int  1 2 6 9 16 18 21 25 26 28 ...
##   ..- attr(*, "names")= chr  "1" "2" "6" "9" ...
```

```
#test data
test_size=training_size
idx_remain = !(1:nrow(autos) %in% idx_train)
autos_remain = autos[idx_remain, ]
idx_test=sample(1:nrow(autos_remain),test_size)
autos_test= autos_remain[idx_test,]
autos_test[,columns_factor]=lapply(autos_test[,columns_factor], as.factor)
rm(autos_remain)
```

2.1 Method 1 - a straightforward but dump approach

In this first approach to finding a candidate model, we use an additive model using all available predictors as the starting model in BIC backward search. Once we have the chosen model, we will examine the LINE model assumptions using a fitted-verse-residual plot model and a Q-Q plot.

2.1.1 Start with an additive model using all predictors (numeric + factor)

```
model1_start=lm(price~.,data=autos_train)
#size of staring model
length( coef(model1_start) )
```

```
## [1] 20
```

2.1.2 Run a backward BIC search

```
#Run BIC backward search
n=nrow(autos_train)
model1_bic = step(model1_start,k=log(n),trace = 0)
#size of selected model by backward BIC
length( coef(model1_bic) )
```

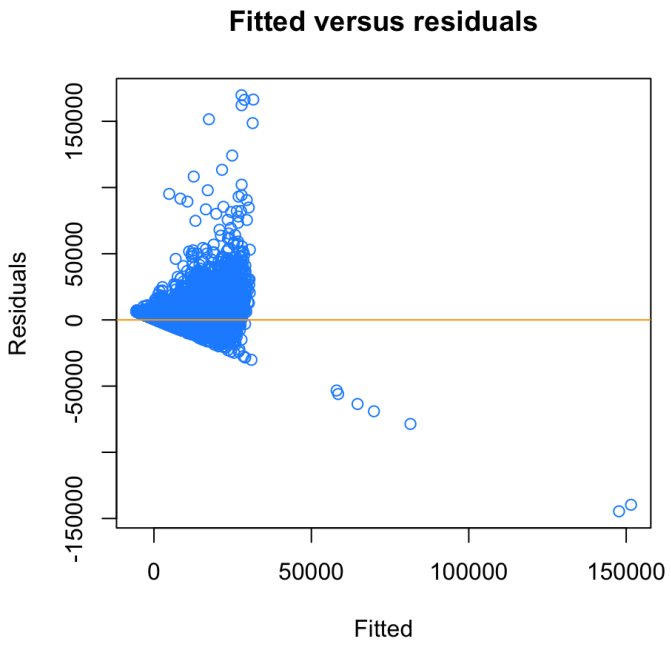
```
## [1] 19
```

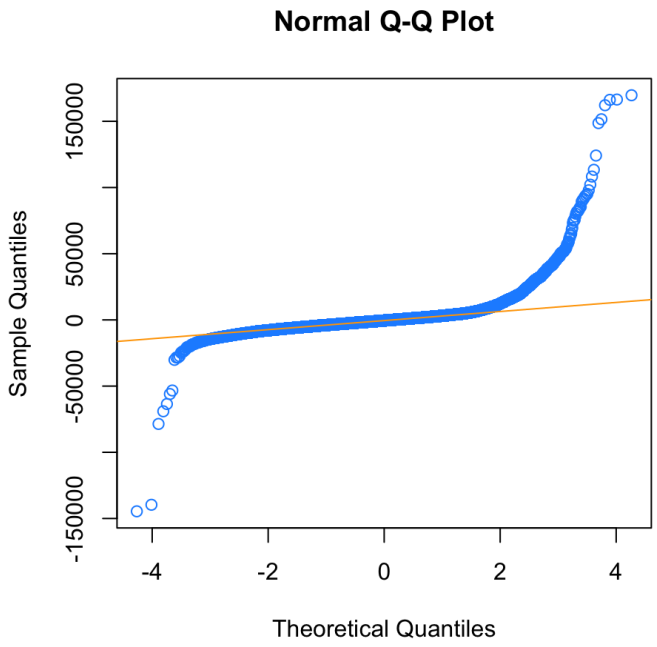
```
model1_bic
```

```
##
## Call:
## lm(formula = price ~ vehicleType + yearOfRegistration + gearbox +
##     powerPS + kilometer + fuelType + notRepairedDamage, data = autos_train)
##
## Coefficients:
##           (Intercept)         vehicleTypebus      vehicleTypecabrio
##             -1.89e+06              5.07e+02              4.33e+03
##      vehicleTypecoupe  vehicleTypekleinwagen       vehicleTypekombi
##              5.85e+03             -1.94e+03              4.36e+02
##  vehicleTypelimousine         vehicleTypesuv     yearOfRegistration
##              1.06e+03              3.89e+03              9.48e+02
##        gearboxmanuell               powerPS             kilometer
##             -3.67e+03              8.69e+00             -3.83e-02
##        fuelTypebenzin           fuelTypecng        fuelTypediesel
##             -2.33e+02             -1.46e+03              8.43e+02
##       fuelTypeelektro        fuelTypehybrid           fuelTypelpg
##             -8.12e+03             -1.04e+03             -8.99e+02
## notRepairedDamagenein
##              1.98e+03
```

2.1.3 Check for violation of model assumptions

```
diagnostics(model1_bic)
```



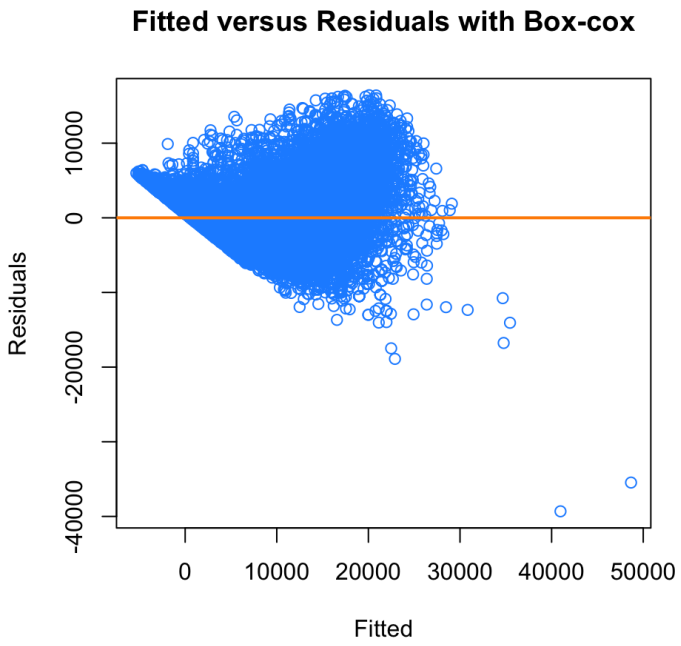


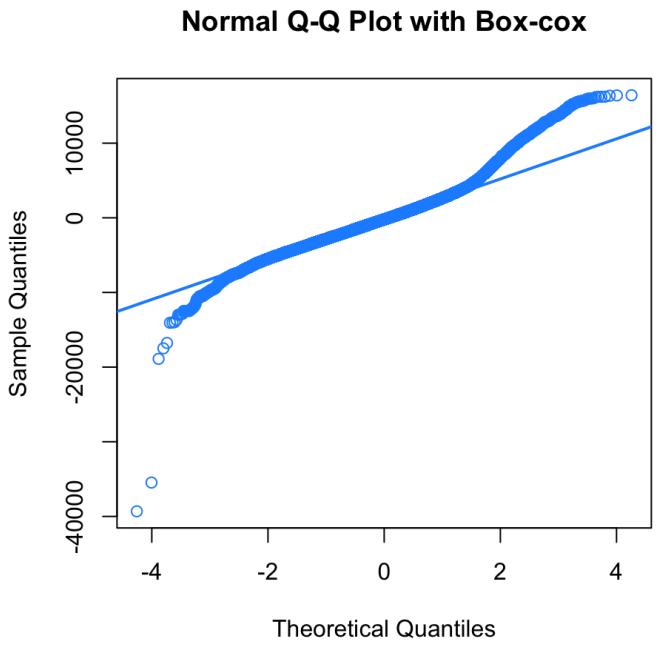
```
#Save this model as the first candidate model
model1_selected=model1_bic
```

Based on the plot results, we concluded this is not a good model as the Fitted Residuals and Q-Q Plots indicated some sort variable tranformations are needed.

2.1.4 Remove high influential points to see improvement in model assumption violations

```
remove_high_influential_points_and_refit_model(model1_selected,autos_train)
```





```
## $removed.n
## [1] 1517
##
## $removed.fraction
## [1] 0.03034
##
## $new.model
##
## Call:
## lm(formula = formula, data = data1, subset = !high_infl)
##
## Coefficients:
##           (Intercept)         vehicleTypebus      vehicleTypecabrio
##             -1.59e+06               1.84e+02               2.13e+03
##      vehicleTypecoupe  vehicleTypekleinwagen       vehicleTypekombi
##              1.63e+03              -3.37e+02              -3.71e+02
##  vehicleTypelimousine         vehicleTypesuv     yearOfRegistration
##              2.37e+02               1.66e+03               7.97e+02
##        gearboxmanuell                powerPS              kilometer
##             -1.11e+03               4.54e+01              -3.35e-02
##           fuelTypecng         fuelTypediesel        fuelTypeelektro
##             -3.73e+02               1.30e+03              -1.41e+03
##        fuelTypehybrid            fuelTypelpg  notRepairedDamagenein
##              3.73e+03              -8.57e+02               1.78e+03
```

Removing the high influential points (about 3% of observations) does improve the model a little. But it is still not ideal. We saved the model (model1_selected) as the first candidate model. In the result section, we will compare its performance with other candidate models.

Because there seems to be a violation of the linearity assumption here, we will introduce response and predictor transformation in the next method.

2.2 Method 2 - With variable transformations

In this second approach to finding a good price prediction model, we will introduces variable transformations. We will take the following 2-phases approach:

- Phase 1 - Use the only continuous variables to determine the response transformation using Box-Cox and the predictor transformation using BIC backward search.
- Phase 2 - Adding categorical variables to the model format obtained from Phase 1.

2.2.1 - Phase 1: determine the best model formula format using only the continuous variables

2.2.1.1 - Break data-set into subgroups by fixing factor variable values

To reduce the price variability due to different combinations of factor variable values, we will temporarily isolate a dataset for the analysis here in phase 1. Later, we will use the full training dataset in Phase 2 steps. To do this, we will use 'dplyr' package count() to group dataset by all factor variables.

```
#listing out all factor columns
columns_factor
```

```
## [1] "abtest"            "vehicleType"       "gearbox"
## [4] "fuelType"          "notRepairedDamage"
```

```
autos_factor_groups=autos %>% count (abtest,vehicleType,gearbox,fuelType,notRepaire
dDamage)
autos_factor_groups=autos_factor_groups[order(autos_factor_groups$n,decreasing = TR
UE),]

#structure of autos_factor_groups
head(autos_factor_groups)
```

```
## # A tibble: 6 x 6
##   abtest  vehicleType gearbox fuelType notRepairedDamage     n
##   <chr>   <chr>       <chr>   <chr>    <chr>              <int>
## 1 test    kleinwagen  manuell benzin   nein               14560
## 2 control kleinwagen  manuell benzin   nein               13311
## 3 test    limousine   manuell benzin   nein               10072
## 4 control limousine   manuell benzin   nein                9423
## 5 test    kombi       manuell diesel   nein                7891
## 6 control kombi       manuell diesel   nein                7290
```

```
#Total number of groups
nrow(autos_factor_groups)
```

```
## [1] 269
```

```
#Number of groups with more than 300 records
sum(autos_factor_groups$n>300)
```

```
## [1] 66
```

Next, we will break our dataset into groups, we will choose group #2 to isolate a dataset by filtering from the main dataset.

2.2.1.2 - isolate a subset data (autos_1) for Phase 1

```
#choose group
selected_group_idx=2
group1=autos_factor_groups[selected_group_idx, ]
(group1.size = group1$n)
```

```
## [1] 13311
```

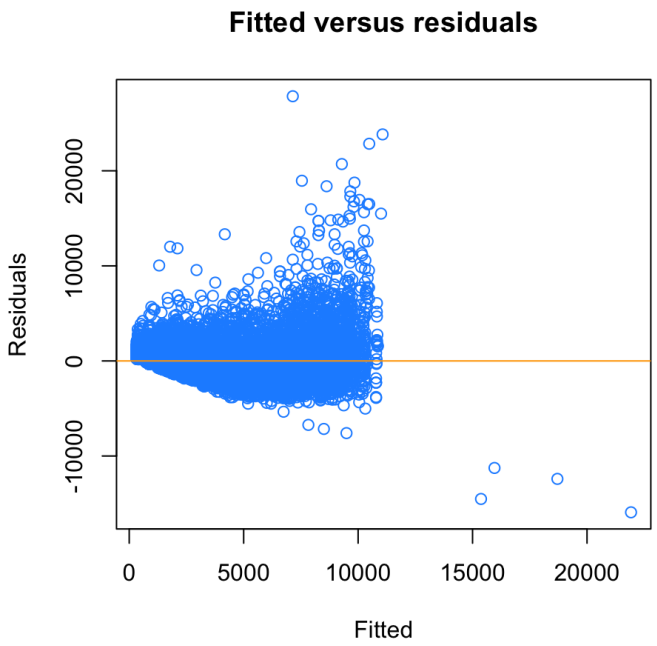
```
group1=subset(group1, select = -c(n) )

#get the records for the selected group
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 )
```

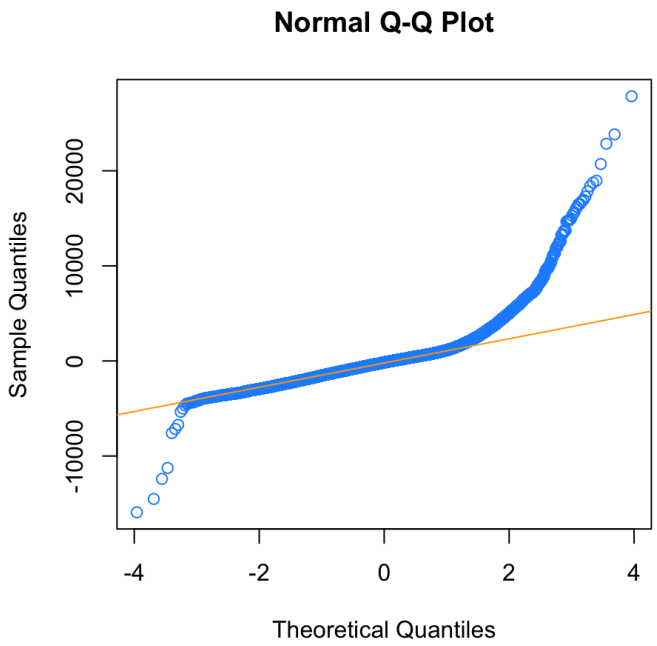
Next, we will use the newly isolated dataset, **autos_1** to help better finding a model formula form for continuous variables

2.2.1.3 - Try an additive model using all continous variables

We will start with fitting an additive model using all continous variables.

```
par(mfrow=c(1,2))
model2_add = lm(price ~ powerPS +yearOfRegistration+kilometer, data = autos_1)
diagnostics(model2_add)
```

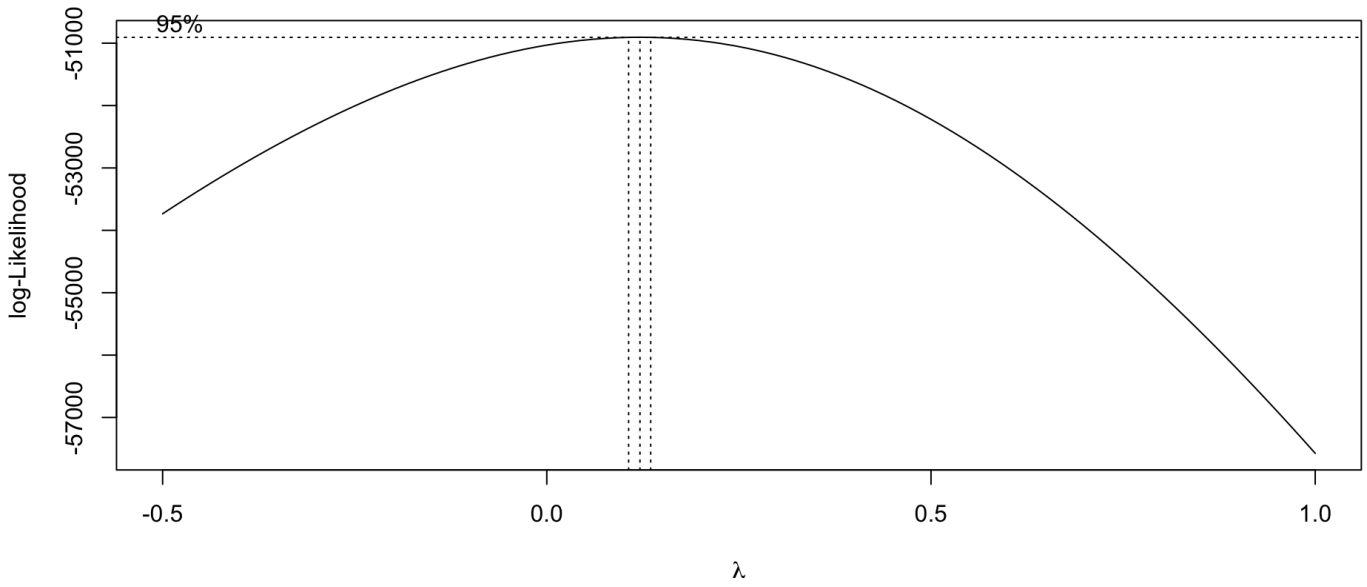




The plots indicates signifant voiloation of equal variance, normality assumption. We will try Box-Cox transformation on Price to improve linearity assumption next.

2.2.1.4 - Try Box-Cox tranformation on the selected group

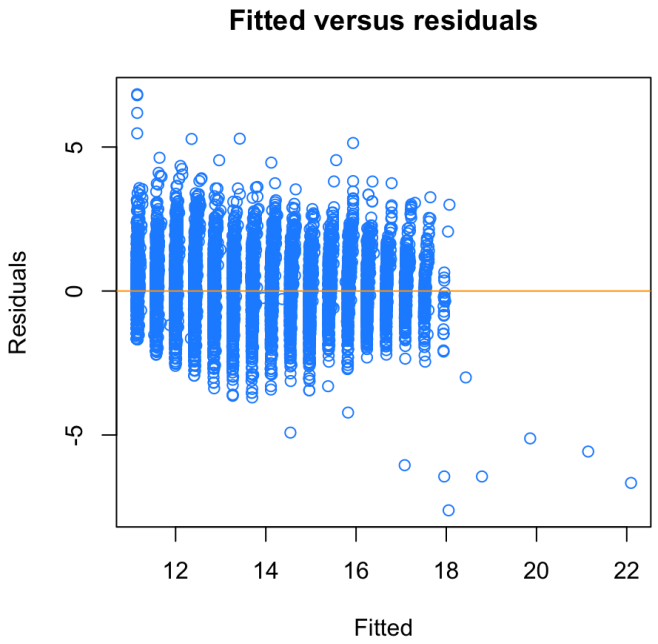
```
par(mfrow=c(1,1))
boxcox_input_formula= as.formula ( as.character(model2_add$call[2]) )
out=boxcox(model2_add, plotit = TRUE, lambda = seq(-0.5, 1.0, by = 0.05))
```

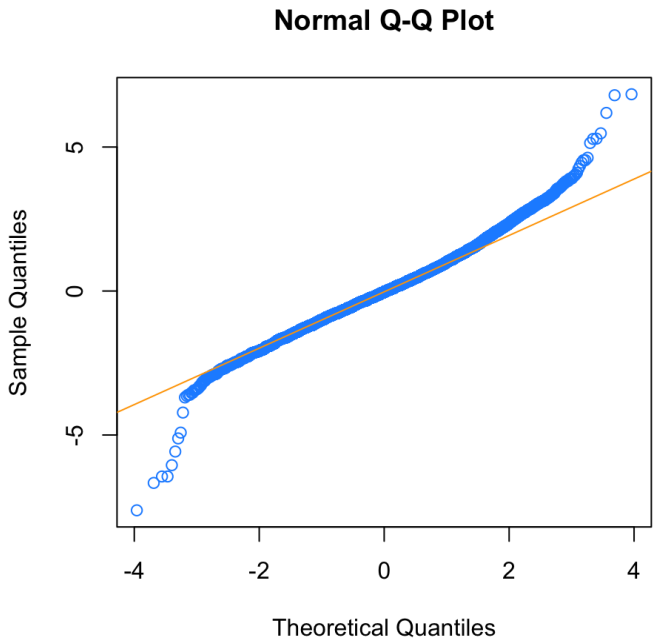



```
( lambda=out$x[which.max(out$y)] )
```

```
## [1] 0.1212
```

```
model2_add_cox = lm( ((price^lambda-1)/lambda) ~ powerPS +yearOfRegistration , data
= autos_1 )
diagnostics(model2_add_cox)
```



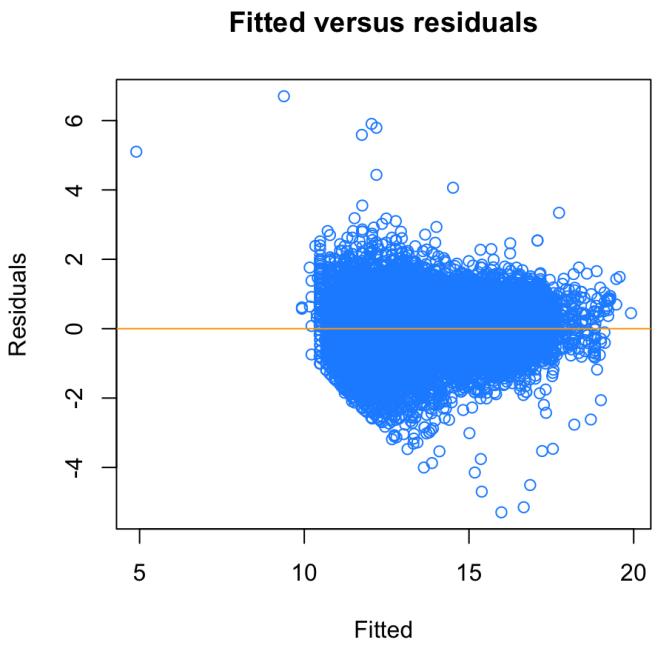


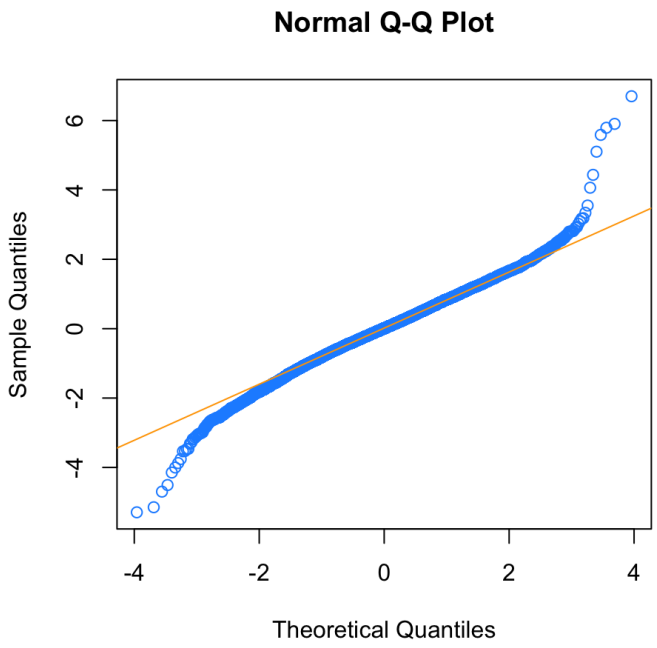
After the Box-Cox transformation (with $\lambda = 0.1212$), the plots look better, but it seems additional predictor transformation might help.

2.2.1.5 - Finding predictor tranformation via backward BIC search

To determine the best predictor transformation, we will use a backward BIC search. We will use a starting model that includes first-order, second-order, and log() terms for all three continuous variables. The resulting BIC search will tell us the best formula format.

```
# starting with log and 2nd order terms for all predictors
model2a_bic_start=lm( (price^lambda-1)/lambda ~ powerPS+I(log(powerPS)) + I(powerPS
^2)
                        +yearOfRegistration + I(log(yearOfRegistration)) + I(yearOf
Registration^2)
                        +kilometer + I(log(kilometer)) + I(kilometer^2)
                      , data = autos_1 )



model2a_bic = step(model2a_bic_start,trace=0)
diagnostics(model2a_bic)
```






```
summary(model2a_bic)
```

```
##
## Call:
## lm(formula = (price^lambda - 1)/lambda ~ powerPS + I(log(powerPS)) +
##     I(powerPS^2) + I(log(yearOfRegistration)) + kilometer + I(log(kilometer)),
##     data = autos_1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.293 -0.528  0.009  0.562  6.700
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -4.58e+03   4.22e+01 -108.55  < 2e-16 ***
## powerPS                    -3.32e-03   3.61e-04   -9.20  < 2e-16 ***
## I(log(powerPS))             2.62e+00   4.73e-02   55.36  < 2e-16 ***
## I(powerPS^2)                2.10e-07   4.41e-08    4.76  1.9e-06 ***
## I(log(yearOfRegistration))  6.03e+02   5.55e+00  108.51  < 2e-16 ***
## kilometer                  -1.50e-05   5.03e-07  -29.83  < 2e-16 ***
## I(log(kilometer))           3.79e-01   3.31e-02   11.48  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.865 on 13304 degrees of freedom
## Multiple R-squared:  0.823,  Adjusted R-squared:  0.823
## F-statistic: 1.03e+04 on 6 and 13304 DF,  p-value: <2e-16
```

```
#Save model2a for later steps
model2a=model2a_bic
formula_str=as.character(model2a$call[2])
```

The best formula format determined by backward BIC search(using only continuous variables) is:

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

Next, we will use this formula form plus categorical variables for finding model in Phase 2.

2.2.2 - Phase 2: adding categorical variables to the Phase-1 model formula

From Phase 1, we determine the best model based on all continous variables only(ie. without categorial column) is:
- (price^lambda - 1)/lambda ~ powerPS + I(log(powerPS)) + I(powerPS^2) + I(log(yearOfRegistration)) + kilometer + I(log(kilometer))
- Where we will determine the new λ value next step, based on entire training dataset

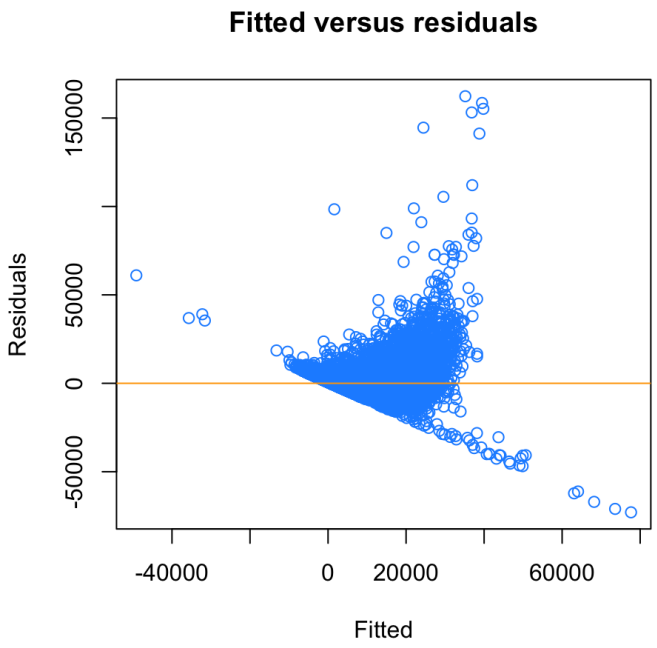
Next, we will add all factor variables to the formula format we obtained from Phase 1 as the starting model in backward BIC search.

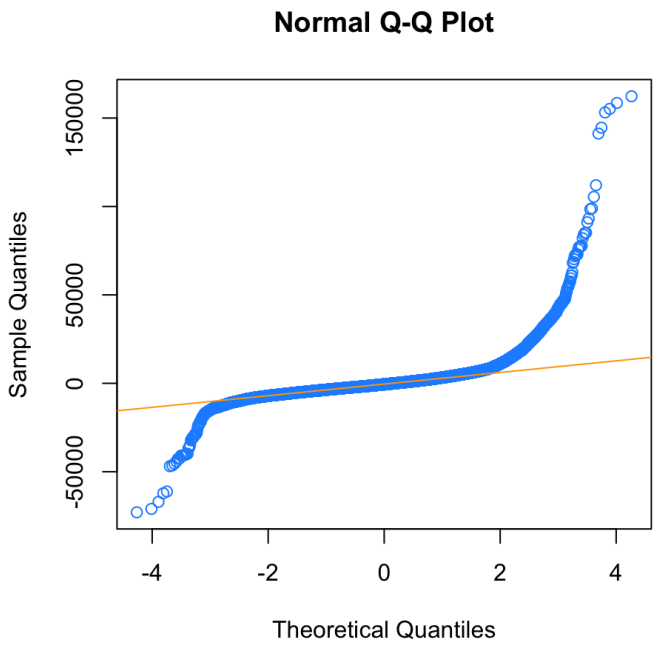
2.2.2.0 - Run box-cox tranformation on all training data

Because the preious λ value was based on a smaller isolated dataset, we will need to re-determine λ value by fitting established model form on all training data.

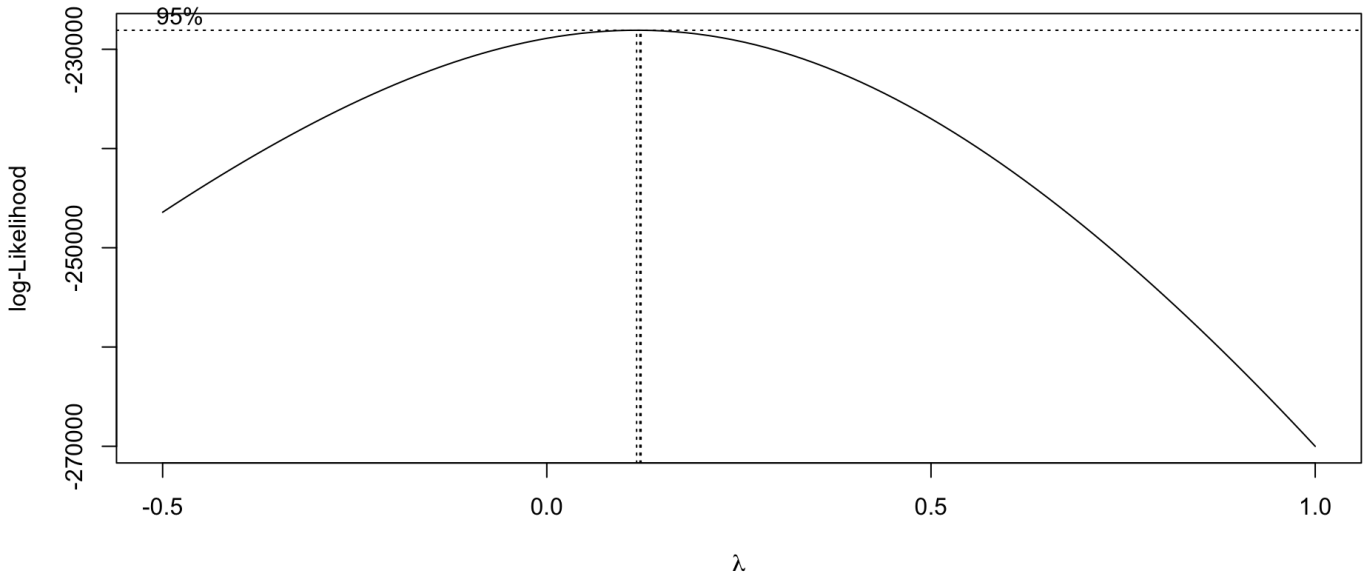
- **Formula format determined from Phase 1**: (price^lambda - 1)/lambda ~ powerPS + I(log(powerPS)) + I(powerPS^2) + I(log(yearOfRegistration)) + kilometer + I(log(kilometer))

```
par(mfrow=c(1,1))

model_2a_all= lm (price ~ powerPS + I(log(powerPS)) + I(powerPS^2) + I(log(yearOfRe
gistration)) + kilometer + I(log(kilometer)),data=autos_train)

diagnostics(model_2a_all)
```






```
par(mfrow=c(1,1))
out=boxcox(model_2a_all, plotit = TRUE, lambda = seq(-0.5, 1.0, by = 0.05))
```




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

```
## [1] 0.1212
```

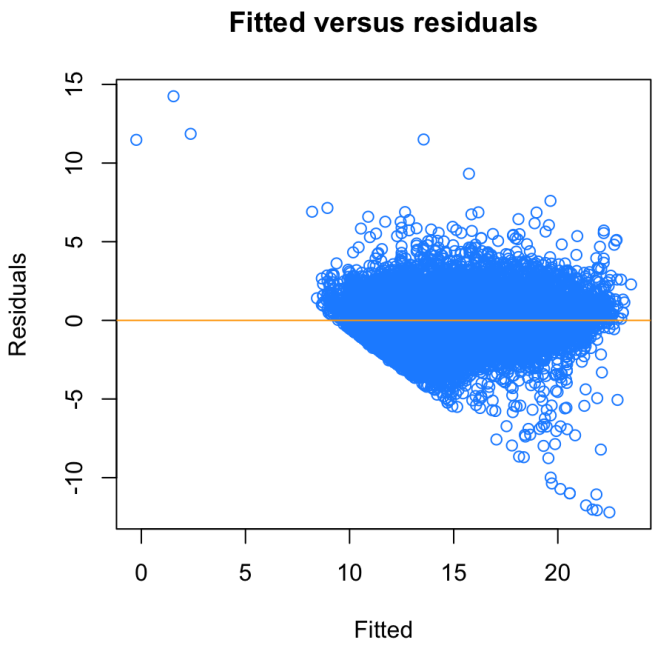
2.2.2.1 Searching for a good model using backward BIC with model format from Phase 1

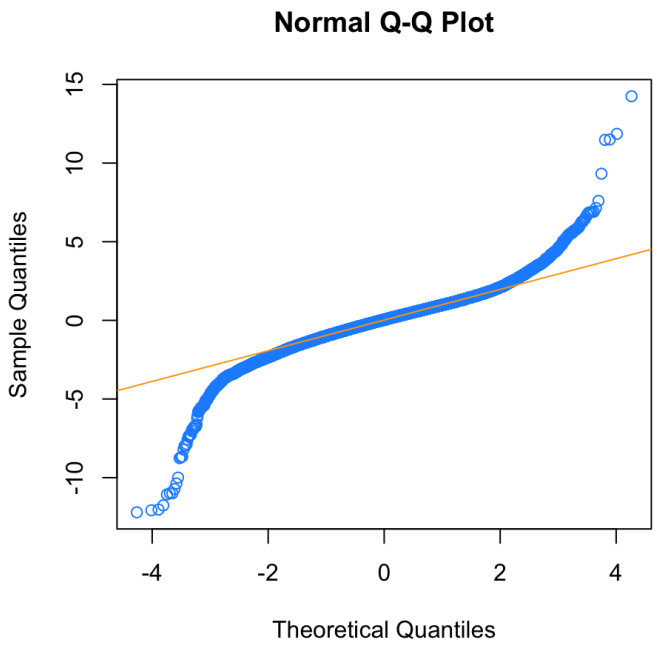
Here, we will find a useful model with a BIC search. We will start a backward search with the model, including all predictors(numeric plus categorial), and those transformation terms that we obtained from Phase 1.

```
par(mfrow=c(1,1))
#model2_start = lm( ((price^lambda - 1)/lambda) ~ .-vehicleType +  I(powerPS^2), da
ta=autos_train )
model2_start = lm( ((price^lambda - 1)/lambda) ~ .+I(log(powerPS)) + I(powerPS^2) +
I(log(yearOfRegistration)) + I(log(kilometer)), data=autos_train )

n=nrow(autos_train)
model2_selected_bic = step(model2_start,k=log(n),trace=0)
summary(model2_selected_bic)
```

```
##
## Call:
## lm(formula = ((price^lambda - 1)/lambda) ~ vehicleType + gearbox +
##     powerPS + kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer)), data = autos_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.205  -0.638   0.044   0.678  14.248
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -5.48e+03   2.70e+01 -202.79  < 2e-16 ***
## vehicleTypebus              8.50e-02   7.04e-02    1.21  0.22736
## vehicleTypecabrio           9.07e-01   7.17e-02   12.66  < 2e-16 ***
## vehicleTypecoupe            5.28e-01   7.29e-02    7.25  4.3e-13 ***
## vehicleTypekleinwagen      -2.56e-02   7.00e-02   -0.36  0.71517
## vehicleTypekombi           -2.48e-01   6.97e-02   -3.56  0.00037 ***
## vehicleTypelimousine        5.94e-02   6.96e-02    0.85  0.39293
## vehicleTypesuv              4.82e-01   7.23e-02    6.67  2.7e-11 ***
## gearboxmanuell             -3.62e-01   1.27e-02  -28.56  < 2e-16 ***
## powerPS                    -1.15e-03   4.24e-05  -27.08  < 2e-16 ***
## kilometer                  -1.60e-05   3.74e-07  -42.84  < 2e-16 ***
## fuelTypebenzin             -1.16e+00   6.42e-01   -1.81  0.07004 .
## fuelTypecng                -1.23e+00   6.51e-01   -1.89  0.05850 .
## fuelTypediesel             -7.81e-01   6.42e-01   -1.22  0.22388
## fuelTypeelektro             1.06e+00   7.54e-01    1.41  0.15832
## fuelTypehybrid             -5.90e-01   6.58e-01   -0.90  0.37061
## fuelTypelpg                -1.33e+00   6.44e-01   -2.06  0.03921 *
## notRepairedDamagenein       1.31e+00   1.93e-02   67.67  < 2e-16 ***
## I(log(powerPS))             2.85e+00   1.87e-02  152.50  < 2e-16 ***
## I(log(yearOfRegistration))  7.20e+02   3.55e+00  202.63  < 2e-16 ***
## I(log(kilometer))           5.50e-01   2.49e-02   22.14  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.11 on 49979 degrees of freedom
## Multiple R-squared:  0.825,  Adjusted R-squared:  0.825
## F-statistic: 1.18e+04 on 20 and 49979 DF,  p-value: <2e-16
```

```
diagnostics(model2_selected_bic)
```






```
model2_selected=model2_selected_bic

model2_selected_bic$call[2]
```

```
## (((price^lambda - 1)/lambda) ~ vehicleType + gearbox + powerPS +
##     kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer)))()
```

We saved the chosen BIC model, model2_selected as our second candidate model, which will be tested in the result section.

2.2.2.2 Verify regression significant with ANOVA test on high p-Value betas(vehicleType and fuelType)

The model summary coefficient(from the last step) shows some parameters with high p-Value, which is an indication of non-significance. Hence, we will conduct ANOVA tests to verify.

First, we will try to see if “vehicleType” categorical variable is significant.

```
#-vehicleType
null_model_str = paste (as.character(model2_selected_bic$call[2]), "-vehicleType")
model_null= lm(null_model_str, data = autos_train)
anova(model_null,model2_selected_bic)
```

```
## Analysis of Variance Table
##
## Model 1: ((price^lambda - 1)/lambda) ~ vehicleType + gearbox + powerPS +
##     kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer)) - vehicleType
## Model 2: ((price^lambda - 1)/lambda) ~ vehicleType + gearbox + powerPS +
##     kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer))
##   Res.Df   RSS Df Sum of Sq   F Pr(>F)
## 1  49986 66008
## 2  49979 61852  7      4156 480 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The lower p-value assures the “vehicleType” is significant. So we keep it in the model.

Likewise, we will try ANOVA test on “fuelType”

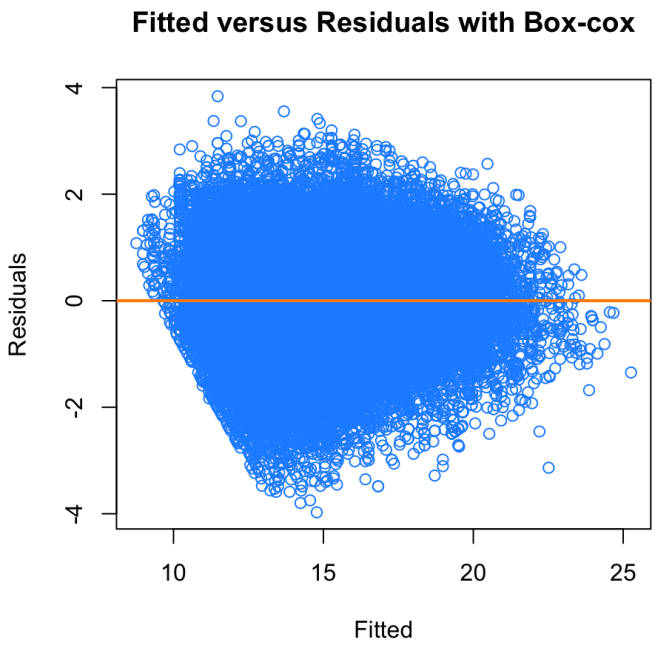
```
#-fuelType
null_model_str = paste (as.character(model2_selected_bic$call[2]), "-fuelType")
model_null= lm(null_model_str, data = autos_train)
anova(model_null,model2_selected_bic)
```

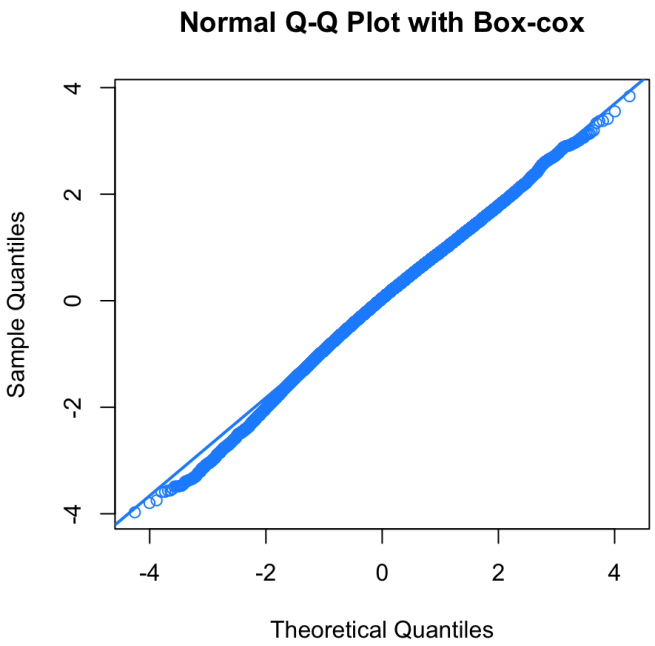
```
## Analysis of Variance Table
##
## Model 1: ((price^lambda - 1)/lambda) ~ vehicleType + gearbox + powerPS +
##     kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer)) - fuelType
## Model 2: ((price^lambda - 1)/lambda) ~ vehicleType + gearbox + powerPS +
##     kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer))
##   Res.Df   RSS Df Sum of Sq   F Pr(>F)
## 1  49985 63181
## 2  49979 61852  6      1329 179 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Again,The lower p-value assures the “fuelType” is aslo significant. So we keep it in the model as well.

2.2.2.3 Remove high influential observations and refit the Phase 2 model

```
fix_model=remove_high_influential_points_and_refit_model(model2_selected_bic, autos
_train)
```


Fitted versus Residuals with Box-cox
Residuals
Fitted
10
15
20
25
-4
-2
0
2
4


Normal Q-Q Plot with Box-cox
Sample Quantiles
Theoretical Quantiles
-4
-2
0
2
4

```
fix_model
```

```
## $removed.n
## [1] 1965
##
## $removed.fraction
## [1] 0.0393
##
## $new.model
##
## Call:
## lm(formula = formula, data = data1, subset = !high_infl)
##
## Coefficients:
##                 (Intercept)              vehicleTypebus
##                   -5.58e+03                    6.96e-02
##           vehicleTypecabrio            vehicleTypecoupe
##                    8.23e-01                    4.25e-01
##      vehicleTypekleinwagen            vehicleTypekombi
##                   -1.31e-01                   -2.16e-01
##       vehicleTypelimousine              vehicleTypesuv
##                    8.29e-02                    4.43e-01
##              gearboxmanuell                     powerPS
##                   -3.32e-01                    6.05e-03
##                   kilometer                fuelTypecng
##                   -1.60e-05                   -2.99e-02
##             fuelTypediesel              fuelTypehybrid
##                    4.05e-01                    5.43e-01
##                fuelTypelpg       notRepairedDamagenein
##                   -1.41e-01                    1.22e+00
##            I(log(powerPS))  I(log(yearOfRegistration))
##                    1.84e+00                    7.34e+02
##          I(log(kilometer))
##                    5.82e-01
```

After removing all high influential observations, The resulting model shows a very lovely Fitted-Residual and Normal Q-Q plot. The high influential observations account for 3.93% of the entire training dataset. The output indicates the chosen model is a good one for the majority(96.07%) of the training data.

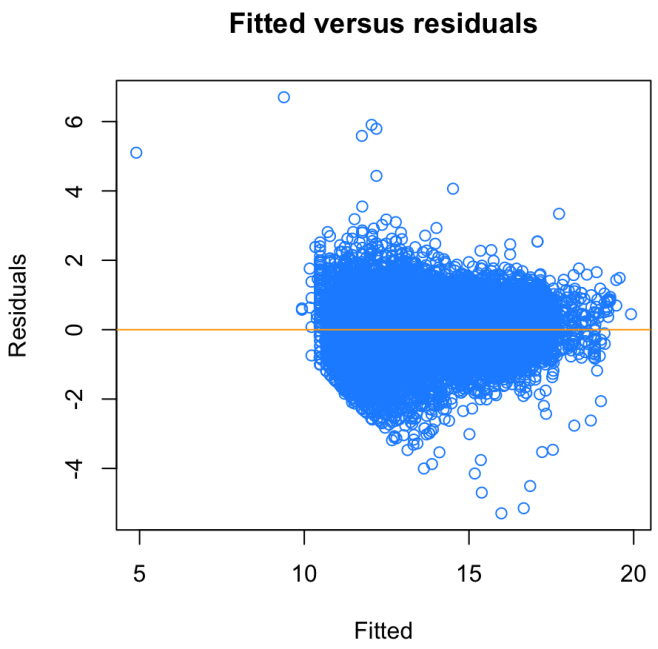
2.3 Method 3 - adding interaction terms on top of method 2 model (Did not yield a new candidate model)

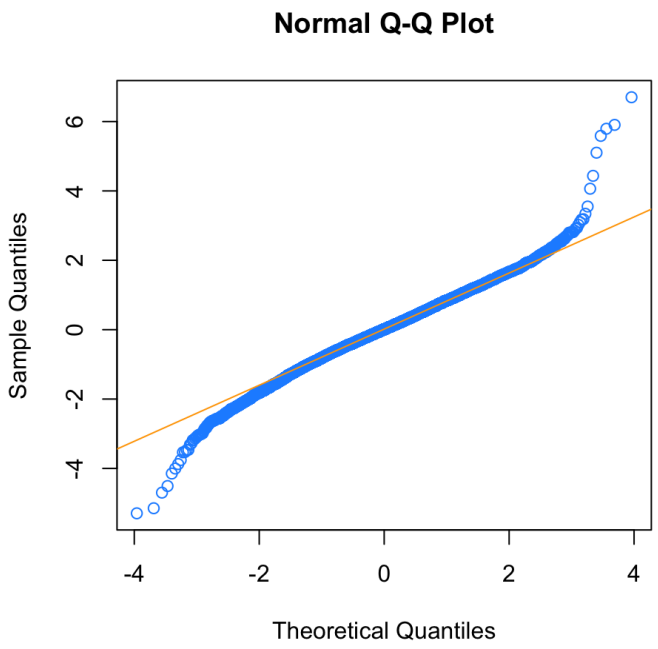
In this method, we will start with the chosen model from Method 2 and see if adding two-way interaction terms will result in a new candidate model, using backward BIC search.

```
library(tictoc)
lambda
```

```
## [1] 0.1212
```

```
#Please note that a two-way interaction term (.^2) is used.
model3a_bic_start=lm( (price^lambda-1)/lambda ~ .^2
                      + I(log(powerPS)) + I(powerPS^2)
                      + I(log(yearOfRegistration)) + I(yearOfRegistration^2)
                      + I(log(kilometer)) + I(kilometer^2)
                      , data = autos_1 )

model3a_bic = step(model2a_bic_start,trace=0)
diagnostics(model3a_bic)
```






```
summary(model3a_bic)
```

```
##
## Call:
## lm(formula = (price^lambda - 1)/lambda ~ powerPS + I(log(powerPS)) +
##     I(powerPS^2) + I(log(yearOfRegistration)) + kilometer + I(log(kilometer)),
##     data = autos_1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -5.293 -0.528  0.009  0.562  6.700
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -4.58e+03   4.22e+01 -108.55  < 2e-16 ***
## powerPS                    -3.32e-03   3.61e-04   -9.20  < 2e-16 ***
## I(log(powerPS))             2.62e+00   4.73e-02   55.36  < 2e-16 ***
## I(powerPS^2)                2.10e-07   4.41e-08    4.76  1.9e-06 ***
## I(log(yearOfRegistration))  6.03e+02   5.55e+00  108.51  < 2e-16 ***
## kilometer                  -1.50e-05   5.03e-07  -29.83  < 2e-16 ***
## I(log(kilometer))           3.79e-01   3.31e-02   11.48  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.865 on 13304 degrees of freedom
## Multiple R-squared:  0.823,  Adjusted R-squared:  0.823
## F-statistic: 1.03e+04 on 6 and 13304 DF,  p-value: <2e-16
```

The resulting model here is the same as the one found in Method 2 Phase 1 (model2a_bic). Therefore, **we won't be able to find a new model** with in this method So, we will stop here.

3. Results

Method 1 resulted in *additive model* which has price as the response with predictors as vehicleType, yearOfRegistration, gearbox, powerPS, kilometer, fuelType and notRepairedDamage.

Method 2 resulted in *interaction model* where response variable price was transformed using boxcox transformation. Predictors used in the model include vehicleType, gearbox, powerPS, kilometer, fuelType, notRepairedDamage and yearOfRegistration. Predictors powerPS, yearOfRegistration and kilometer are log transformed.

We use *train and test RMSEs* to evaluate the chosen 2 models. Please note that since the interaction model response was transformed, we will have to change it back to its unit of measure by applying reverse transformation.

```
#mod1 RMSEs
Train_RMSE_mod1 = calc_rmse(autos_train$price, predict(model1_selected, autos_trai
n))
Test_RMSE_mod1 =  calc_rmse(autos_test$price, predict(model1_selected, autos_test))

#mod2 RMSEs
Train_RMSE_mod2 = calc_rmse(autos_train$price, (predict(model2_selected, newdata =
 autos_train)*lambda+1)^(1/lambda))
Test_RMSE_mod2 = calc_rmse(autos_test$price, (predict(model2_selected, newdata = au
tos_test)*lambda+1)^(1/lambda))

# calculate all train errors
train_error = c(Train_RMSE_mod1, Train_RMSE_mod2)

# calculate all test errors
test_error = c(Test_RMSE_mod1, Test_RMSE_mod2)

auto_models = c("Additive model", "Transformation model")
auto_results = data.frame(auto_models, train_error, test_error)
colnames(auto_results) = c("Model", "Train RMSE", "Test RMSE")
knitr::kable(auto_results)
```

Model	Train RMSE	Test RMSE
Additive model	5883	5960
Transformation model	4345	4493

As seen from the above table, the transformation model has a lower test and train RMSEs, suggesting it to be a better model.

We will now compare the 2 models using coefficient of determination R2, adjusted R2:

```
summary(model1_selected)
```

```
##
## Call:
## lm(formula = price ~ vehicleType + yearOfRegistration + gearbox +
##     powerPS + kilometer + fuelType + notRepairedDamage, data = autos_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -144574   -2781    -477    1850  169687
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -1.89e+06   1.88e+04 -100.56  < 2e-16 ***
## vehicleTypebus          5.07e+02   3.72e+02    1.36    0.173
## vehicleTypecabrio       4.33e+03   3.78e+02   11.44  < 2e-16 ***
## vehicleTypecoupe        5.85e+03   3.84e+02   15.25  < 2e-16 ***
## vehicleTypekleinwagen -1.94e+03   3.69e+02   -5.26  1.4e-07 ***
## vehicleTypekombi        4.36e+02   3.68e+02    1.18    0.236
## vehicleTypelimousine    1.06e+03   3.68e+02    2.88    0.004 **
## vehicleTypesuv          3.89e+03   3.81e+02   10.21  < 2e-16 ***
## yearOfRegistration      9.48e+02   9.19e+00  103.17  < 2e-16 ***
## gearboxmanuell         -3.67e+03   6.32e+01  -58.01  < 2e-16 ***
## powerPS                 8.69e+00   1.94e-01   44.81  < 2e-16 ***
## kilometer              -3.83e-02   8.83e-04  -43.36  < 2e-16 ***
## fuelTypebenzin         -2.33e+02   3.40e+03   -0.07    0.945
## fuelTypecng            -1.46e+03   3.44e+03   -0.43    0.671
## fuelTypediesel          8.43e+02   3.40e+03    0.25    0.804
## fuelTypeelektro        -8.12e+03   3.98e+03   -2.04    0.042 *
## fuelTypehybrid         -1.04e+03   3.48e+03   -0.30    0.766
## fuelTypelpg            -8.99e+02   3.40e+03   -0.26    0.792
## notRepairedDamagenein   1.98e+03   1.02e+02   19.37  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5880 on 49981 degrees of freedom
## Multiple R-squared:  0.545,  Adjusted R-squared:  0.545
## F-statistic: 3.33e+03 on 18 and 49981 DF,  p-value: <2e-16
```

```
summary(model1_selected)$adj.r.squared
```

```
## [1] 0.5452
```

Since this is an additive model, several interpretations can be made from the summary results. As the year of registration increase by one, the price increases by 900 euros. As the kilometers driven by the car increases by one, the price decreases slightly by 0.3829 euros, which sounds correct.

```
#Calculating R2 for additive model
#Total sum of squares
SST = sum((autos_train$price - mean(autos_train$price))^2)

#Residual sum of squares
fitted = predict(model1_selected, newdata = autos_train)
SSReg = sum((fitted - mean(autos_train$price))^2)
(R2 = SSReg / SST)
```

```
## [1] 0.5454
```

```
summary(model2_selected)
```

```
##
## Call:
## lm(formula = ((price^lambda - 1)/lambda) ~ vehicleType + gearbox +
##     powerPS + kilometer + fuelType + notRepairedDamage + I(log(powerPS)) +
##     I(log(yearOfRegistration)) + I(log(kilometer)), data = autos_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.205  -0.638   0.044   0.678  14.248
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -5.48e+03   2.70e+01 -202.79  < 2e-16 ***
## vehicleTypebus              8.50e-02   7.04e-02    1.21  0.22736
## vehicleTypecabrio           9.07e-01   7.17e-02   12.66  < 2e-16 ***
## vehicleTypecoupe            5.28e-01   7.29e-02    7.25  4.3e-13 ***
## vehicleTypekleinwagen      -2.56e-02   7.00e-02   -0.36  0.71517
## vehicleTypekombi           -2.48e-01   6.97e-02   -3.56  0.00037 ***
## vehicleTypelimousine        5.94e-02   6.96e-02    0.85  0.39293
## vehicleTypesuv              4.82e-01   7.23e-02    6.67  2.7e-11 ***
## gearboxmanuell             -3.62e-01   1.27e-02  -28.56  < 2e-16 ***
## powerPS                    -1.15e-03   4.24e-05  -27.08  < 2e-16 ***
## kilometer                  -1.60e-05   3.74e-07  -42.84  < 2e-16 ***
## fuelTypebenzin             -1.16e+00   6.42e-01   -1.81  0.07004 .
## fuelTypecng                -1.23e+00   6.51e-01   -1.89  0.05850 .
## fuelTypediesel             -7.81e-01   6.42e-01   -1.22  0.22388
## fuelTypeelektro             1.06e+00   7.54e-01    1.41  0.15832
## fuelTypehybrid             -5.90e-01   6.58e-01   -0.90  0.37061
## fuelTypelpg                -1.33e+00   6.44e-01   -2.06  0.03921 *
## notRepairedDamagenein       1.31e+00   1.93e-02   67.67  < 2e-16 ***
## I(log(powerPS))             2.85e+00   1.87e-02  152.50  < 2e-16 ***
## I(log(yearOfRegistration))  7.20e+02   3.55e+00  202.63  < 2e-16 ***
## I(log(kilometer))           5.50e-01   2.49e-02   22.14  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.11 on 49979 degrees of freedom
## Multiple R-squared:  0.825,  Adjusted R-squared:  0.825
## F-statistic: 1.18e+04 on 20 and 49979 DF,  p-value: <2e-16
```

```
summary(model2_selected)$adj.r.squared
```

```
## [1] 0.8248
```

```
#Calculating R2 for transformation model
fitted = (predict(model2_selected, newdata = autos_train)*lambda+1)^(1/lambda)

#Residual sum of squares
SSReg = sum((fitted - mean(autos_train$price))^2)

(R2 = SSReg / SST)
```

```
## [1] 0.6202
```

	R2	Adjusted R2	Comment
Additive model	0.5454	0.5452	Preferred for relationship explanation
Transformation model	0.6202	0.8248	Preferred for price prediction

Transformation model has a higher R2 than the Additive model, however that can be the case because transformation is a bigger model with 21 coefficients, compared to additive model that has 19 coefficients. But by checking adjusted R2, we can confirm that transformation model indeed is a preferred model.

4. Discussion

Using a dataset from Kaggle that contains German eBay used car ads, we started by doing some of the data-cleaning to remove unreasonable data from the dataset. We refrained from doing any translations or conversions of data from the German language to English due to our limited time for this project. Furthermore, We created one training dataset consists of 50,000 randomly sampled records, and another testing dataset the same size. Moreover, we tried three different modeling approaches and generated two candidate models. Below are the following methods that we have explored:

In Method 1, we tried fitting an Additive Model with all variables. Using BIC backward search, found the best model where chosen predictors were vehicleType, yearOfRegistration, gearbox, powerPS, kilometer, fuelType and notRepairedDamage. This model violated the linearity, constant variance assumptions, and the normality assumption was also violated, as seen from the Q-Q plot. With this method, we generate a candidate model, which we referred to as the Additive Model.

In Method 2, We introduced variable transformations with the following 2-phases approach:

- Phase 1 - Use the only continuous variables to determine the response transformation using Box-Cox and the predictor transformation using BIC backward search.
- Phase 2 - Adding categorical variables to the model format obtained from Phase 1.

The chosen model in Method 2 used Box-Cox to transform the response(Price). The model includes the following predictors: vehicleType, yearOfRegistration, gearbox, powerPS, kilometer, fuelType, and notRepairedDamage. It also included non-linear terms of log(powerPS), powerPS^2, log(yearOfRegistration), and log(kilometer). We referred to this model as the Transformation Model.

In Method 3, We started with the chosen model from Method 2 and added two-way interaction terms to find a new candidate model using a backward BIC search. However, this method yielded the same results as Method 2.

The results section shows the comparison between the Additive and Transformation models by calculating their test and training RMSEs. Based on both models' results, the RMSE is only slightly higher than the training RMSE, which indicates that we do not have an overfitting issue with the model. The Transformation model test and training RMSEs were 25% to 30% lower than the additive model showing that the transformation model is our preferred model for price prediction. However, we prefer the Additive model for explaining the relationship between the price and predictors. Because it is simpler and does not have any variable transformations, it doesn't include additional non-linear terms.

5. Appendix

Below file lists all the functions being used in the methods section:

```
cat -n misc_functions.R
```

```
#Function to find the best transformation of the form considered by the Box-Cox method
get_boxcox_lambda = function(model1){
  boxcot_out=boxcox(model1, plotit = FALSE, lambda = seq(-0.5, 1.0, by = 0.05))
  lambda=boxcot_out$x[which.max(boxcot_out$y)]
  return(lambda)
}

#Function to remove influential points
remove_high_influential_points_and_refit_model = function (model, data1){
  ret = list()
  #finding influenctial
  cd = cooks.distance(model)
  n=length(resid(model))
  high_infl = cd > 4 / n
  ret[["removed.n"]]=sum(high_infl)
  ret[["removed.fraction"]]=mean(high_infl)
  #Refit the multiple regression model without any influential points
  formula=as.formula(as.character(model$call[2]))
  model_new = lm(formula, data = data1, subset = !high_infl)
  ret[["new.model"]]=model_new
  par(mfrow=c(1,2))
  plot(fitted(model_new), resid(model_new), col = "dodgerblue",
       xlab = "Fitted", ylab = "Residuals", main = "Fitted versus Residuals with Box-cox")
  abline(h = 0, col = "darkorange", lwd = 2)
  qqnorm(resid(model_new), main = "Normal Q-Q Plot with Box-cox", col = "dodgerblue")
  qqline(resid(model_new), col = "dodgerblue", lwd = 2)
  return(ret)
}

#Function to plot Fitted vs. Residuals and Q-Q plot
diagnostics = function(model, pcol="dodgerblue",lcol="orange",alpha=0.05,plotit=TRUE){
  if (plotit ){
    #fitted vs. residual
    par(mfrow=c(1,2))
    plot( model$fitted.values,
          model$residuals,
          col=pcol,
          xlab="Fitted",
          ylab="Residuals",
          main="Fitted versus residuals" )
    abline(h=0,col=lcol,lwd=1)
```

```
42
43      #QQ plot
44      qqnorm(resid(model),main="Normal Q-Q Plot",col=pcol)
45      qqline(resid(model),col=lcol,lwd=1)
46    }
47  }
48
49  #Function to calculate RMSE
50  calc_rmse  = function(actual, predicted) {
51    sqrt(mean((actual - predicted) ^ 2))
52  }
```