An Introduction to R - Used Car Prices

Lukas Vogt, Philipp Thienel

28 November 2015

Table of content

1. The problem

2. Reproducing the analysis

2.1 Structure of the documents

2.2 Dependencies & required libraries

3. The dataset

3.1 Reading the data

3.2 Cleaning the data

3.3 Adding features to data

4. Descriptive analysis

5. Fitting a linear model

6. Evaluation of fit and predictive power

1. The problem

The following problem was given:

The dataset contains records of a website selling used cars from Jul 2011. Along with the price you find various characteristics of different VW station wagons (Golf, Passat, Bora, Caddy, Multivan).

1. Import data and clean it appropriately
2. Analyze the dataset from a descriptive point of view. What do you observe for prices? How have they potentially been sampled?
3. Find a regression model that has the prices as dependent variable. Look at different models and try to identify variable that you expect to drive the price. Also think about dummies and interaction terms. Construct an additional variable "age", i.e., the difference between the first registration (inverkehrssetzung) of the vehicle and Jul 2011.
4. Which is the most reasonable / best regression model? Does it explain the prices well?
5. Illustrate your findings graphically as well as in tabular and text form.

2. Reproducing the analysis

To reproduce the analysis the reader can just execute all steps as shown by himself. Alternatively (and) under the following url 'https://github.com/philippthienel/hsg-intro-r-a03' a repository can be found including all required scripts, functions and datasets.

2.1 Structure of the documents

Explain how data, files, functions etc. are organized, Where what can be found.

2.2 Dependencies & required libraries

Following R libraries will be required:

- ggplot2
- knitr
- stringr
- reshape2

3. The dataset

The dataset can be downloaded here: 'https://github.com/philippthienel/hsg-intro-r-a03'. According to the problem the dataset "contains records of a website selling used cars from Jul 2011". The dataset of interest is 'vw_station_wagon.csv'. The file 'variables.csv' provides a list of all variable names in the dataset along with english explanations.

```r
variable.names <- data <- read.csv("./data/variables.csv", sep=';')
kable(variable.names)
```

Variable	Description.in.English
modell	model

Variable	Description.in.English
inverkehrssetzung	first registration of the car Month-Year
fahrzeugart	type of car
aussenfarbe	outside color
kilometer	kilometers
getriebeart	transmission type (manual/automatic)
antrieb	front wheel / rear wheel / 4 wheel
treibstoff	fuel type
tueren	number of doors
sitze	number of seats
innenfarbe	inside color
hubrauminccm	engine capacity in ccm
zylinder	cylinder
leistunginps	power in horsepower
leergewicht	curb weight
verbrauch	fuel consumption in liters / 100km
co2.emission	co2 emissions
energieeffizienz	energency efficiency
abmfk	certification of vehicle by authorities, i.e., you can only use a car if you have the MFK
garantie	warranty
preis	price

3.1 Reading and the data

The dataset is contained in the folder 'data'. The file 'A03_read_clean.R' contains all code to read and prepare the data for the analysis.

We will read the dataset and assign the dataframe to 'data':

```
path.data <- './data/vw_station_wagons.csv'
data <- read.csv(path.data, sep=',')
```

3.2 Cleaning the data

We can easily see that the dataset is not in a perfect condition for further analysis, particularly with regards to two points:

1. The names of units are included in the datafields and the variable is therefore of type character, while it should be integer or numeric.

```
index <- c('kilometer', 'verbrauch', 'leergewicht', 'co2.emission', 'garantie', 'preis')
kable(head(data[,index]))
```

kilometer	verbrauch	leergewicht	co2.emission	garantie	preis
26'200 km	8.3 l/100km	1570 kg	198 g/km	12 Monate	CHF 21'400.-
101'500 km	10 l/100km	1790 kg	240 g/km	12 Monate	CHF 24'900.-
113'000 km	6.2 l/100km	1613 kg	167 g/km	0 Monate	CHF 22'800.-
166'000 km	11 l/100km	1627 kg	266 g/km	0 Monate	CHF 7'900.-
133'000 km	86 l/100km	1901 kg	80 g/km	0 Monate	CHF 18'000.-
78'000 km	6.3 l/100km	1385 kg	148 g/km	0 Monate	CHF 18'000.-

To remove the units from the data we will define a function that takes a character vector as input, extracts the first numerical sequence in every element of the input vector and returns these tuples in a numerical output vector of the same length as the input.

```
GetValue <- function(x) {
  require(stringr)
  x <- gsub("'","",x)
  x <- str_extract(x, '[0-9]+\\.*[0-9]*')
  return(as.numeric(x))
}
```

Then we will call this function on all variables of the dataset, that included units, replacing the old values.

```
data <- within(data,{
  kilometer <- GetValue(kilometer)
  leergewicht <- GetValue(leergewicht)
  verbrauch <- GetValue(verbrauch)
  co2.emission <- GetValue(co2.emission)
  garantie <- GetValue(garantie)
  preis <- GetValue(preis)
})
```

As we can see, the variables are now numerical and do not include the units any longer.

```
index <- c('kilometer', 'verbrauch', 'leergewicht', 'co2.emission', 'garantie', 'preis')
kable(head(data[,index]))
```

kilometer	verbrauch	leergewicht	co2.emission	garantie	preis
26200	8.3	1570	198	12	21400
101500	10.0	1790	240	12	24900
113000	6.2	1613	167	0	22800
166000	11.0	1627	266	0	7900
133000	86.0	1901	80	0	18000
78000	6.3	1385	148	0	18000

2. Variables are encoded as integer, while they should rather be treated as factors.

```
kable(sapply(data[,c("tueren","sitze","zylinder")], class))
```

tueren	integer
sitze	integer
zylinder	integer

```
kable(summary(data[,c("tueren","sitze","zylinder")]))
```

tueren	sitze	zylinder
Min. :3.000	Min. :2.000	Min. :4.000
1st Qu.:5.000	1st Qu.:5.000	1st Qu.:4.000

tueren	sitze	zylinder
Median :5.000	Median :5.000	Median :4.000
Mean :4.894	Mean :5.059	Mean :4.088
3rd Qu.:5.000	3rd Qu.:5.000	3rd Qu.:4.000
Max. :5.000	Max. :9.000	Max. :6.000
NA's :2	NA's :57	NA's :7

We can simply cast the respective variables as factors.

```r
data <- within(data,{
  tueren <- as.factor(tueren)
  sitze <- as.factor(sitze)
  zylinder <- as.factor(zylinder)
})
```

3.3 Adding features to the data

Three additional features that can be extracted from the existing variables are:

1. plattform of the model (Golf, Bora, Passat etc.) - can be extracted from the 'modell' variable
2. age of the car in months - we will just take the time difference between first registration and July 2011 (when the data was sampled) as an approximation
3. displacement in litres - can be optained by dividing 'hubrauminccm' by 1000

1. Extracting the 'plattform' from the 'model' variable. The plattform is always the second word in the 'modell' variable. So we need to extract only the second word of every element of that variable.

Noting that the first word is always 'VW' we can define a function 'GetPlattform' that takes a character vector as input, removes any string 'VW' from every element and then extracts always the first character sequence of length >= 1 until the first character that is not a letter.

```r
GetPlattform <- function(x) {
  require(stringr)
  x <- gsub('VW','',x)
  plattform <- str_extract(x, '[a-zA-Z]+')
  return(plattform)
}
```

We call this function on the variable 'modell' to extract the plattform and assign it to the variable 'plattform'

```r
data$plattform <- as.factor(GetPlattform(data$modell))
```

2. Calculating the age of the vehicle We can get the month and year of the first registration from the variable 'inverkehrssetzung'. This variable is of type factor and has the structure: 'month-year', so for example '01-2011'.

To calculate the age we define a function that takes a factor variable as input. Splits the factor by '-' in two columns (month and year) and then calculates the difference between the month of the first registration and July 2011.

```
GetAge <- function(x, month=7, year=2011) {
  require(reshape2)
  df <- colsplit(x,"-",names=c("month","year"))
  age <- (year - df$year)*12 + (month - df$month)
  return(age)
}
```

We call that function on the 'inverkehrssetzung' variable to calculate the approximate age of the vehicle and assign the values to the new variable 'age'.

```
data$age <- GetAge(data$inverkehrssetzung)
```

3. Calculating displacement in liters This is just the simple measure of scaling the variable 'hubrauminccm' down by factor 1000. We will keep one decimalpoint however, since it is customary to indicate the size of the engine like that.

```
data$hubraum.liter <- round(data$hubraum/1000,1)
```

4. Descriptive analysis

All relevant code can be found in 'A03_descriptive.R' and 'A03_explorative.R'.

Steps:

1. Descriptive analysis of dataset
2. Missing values
3. Preis - the dependent variable
4. Numeric covariates of interest
5. Factor variables

4.1 Descriptive analysis of dataset

Looking at the basics, we are primarily interested in the following metrics:

Number of observations and variables

```
dimensions <- dim(data)
names(dimensions) <- c("observations","variables")
kable(dimensions)
```

observations	1170
variables	24

Types of variables Determine class for every variable in the dataset.

```
variables <- sapply(data, class)
```

Numerical variables:

```r
numerical.variables <- variables[variables %in% c("integer","numeric")]
kable(numerical.variables)
```

kilometer	numeric
hubrauminccm	integer
leistunginps	integer
leergewicht	numeric
verbrauch	numeric
co2.emission	numeric
garantie	numeric
preis	numeric
age	numeric
hubraum.liter	numeric

Categorical variables:

```r
categorical.variables <- variables[variables %in% c("factor","logical")]
kable(categorical.variables)
```

modell	factor
inverkehrssetzung	factor
fahrzeugart	factor
aussenfarbe	factor
getriebeart	factor
antrieb	factor
treibstoff	factor
tueren	factor
sitze	factor
innenfarbe	factor
zylinder	factor
energieeffizienz	factor
abmfk	logical
plattform	factor

4.2 Missing values

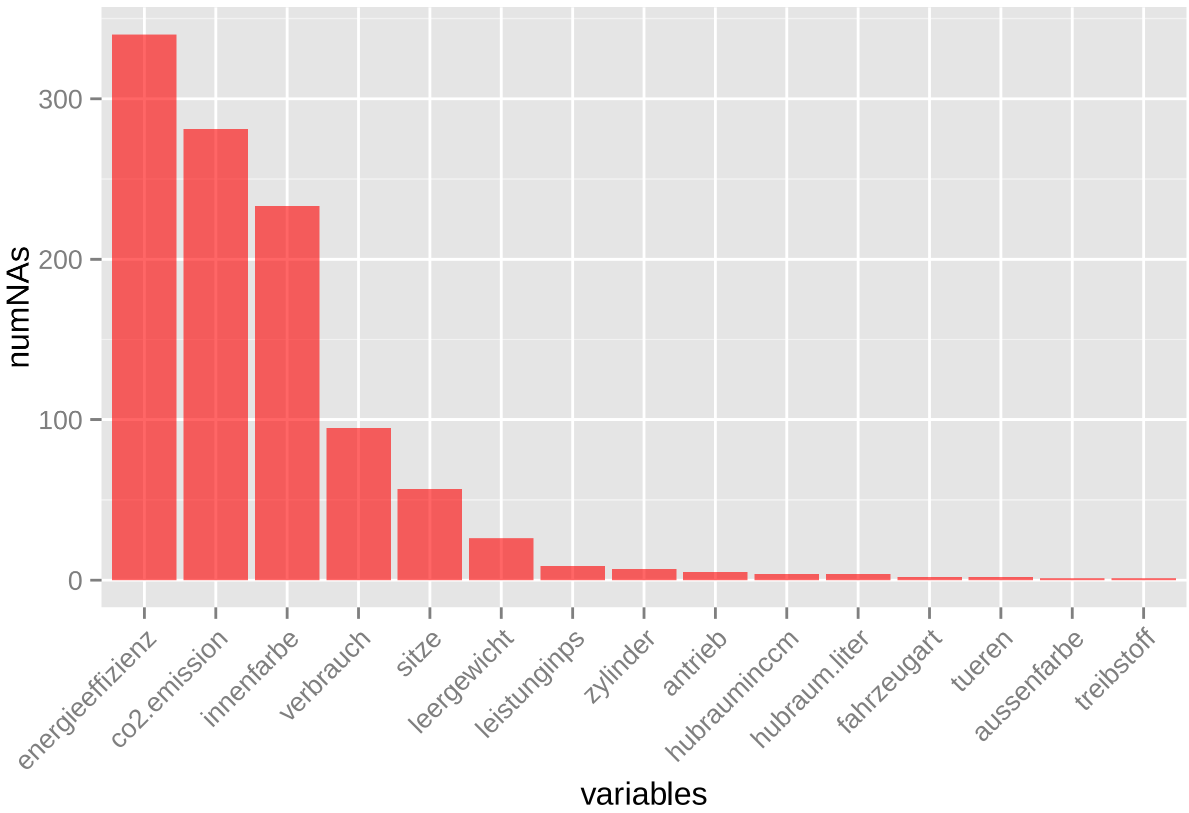
```r
CountNA <- function(df){
  count <- sapply(df, FUN = function(x) sum(is.na(x)))
  count <- data.frame(variable = names(count), count.na = count,
                      row.names=NULL)
  count <- count[count$count.na>0,]
  index <- order(count$count.na, decreasing=T)
  return(count[index,])
}
```

```r
count.na <- CountNA(data)
kable(count.na)
```

	variable	count.na
18	energieeffizienz	340
17	co2.emission	281
11	innenfarbe	233
16	verbrauch	95
10	sitze	57
15	leergewicht	26
14	leistunginps	9
13	zylinder	7
7	antrieb	5
12	hubrauminccm	4
24	hubraum.liter	4
3	fahrzeugart	2
9	tueren	2
4	aussenfarbe	1
8	treibstoff	1

```
source("./functions/naBar.R")
```

```
naBar(data)
```

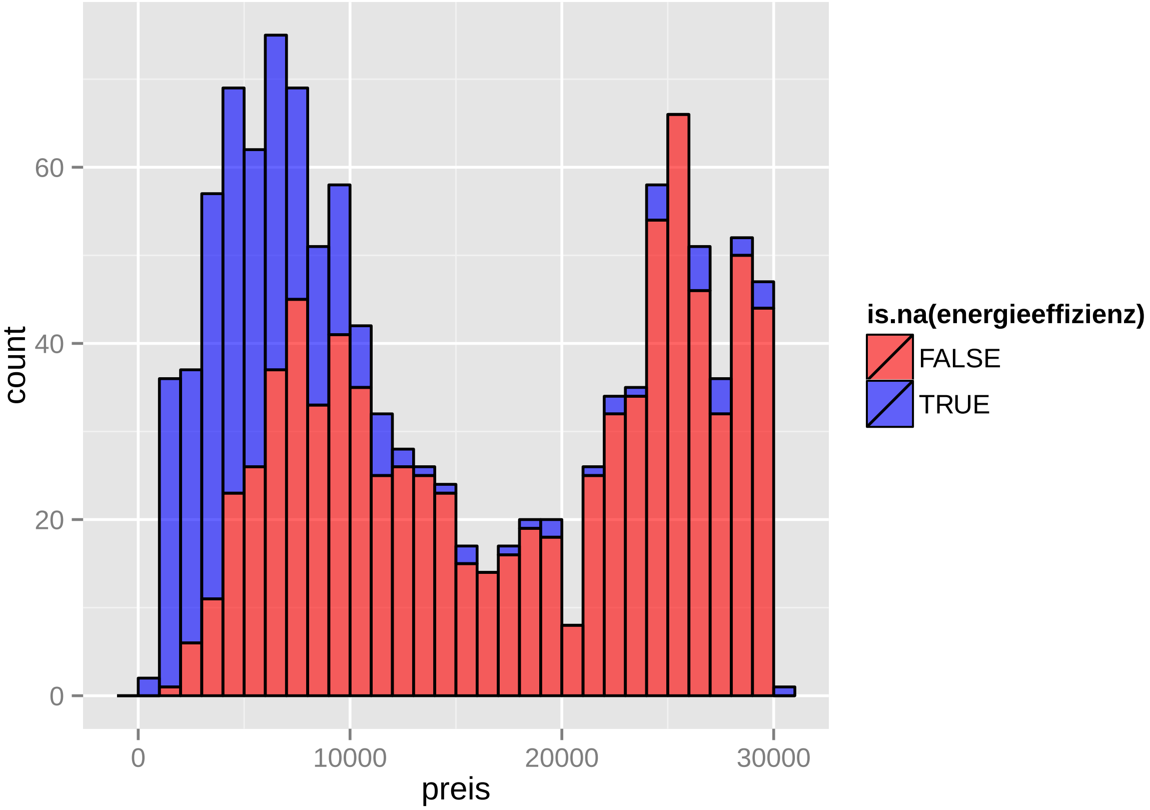


Bias in missing values Would we introduce a selection bias if we exclude the observations where the variable ‘energieeffizienz’ is missing?

Yes, we would. Easy to see, that ‘energieefizienz’ is missing disproportionally in lower ranges of the dependent variable ‘preis’.

If we would use that variable in our regression later, we would have to pay special attention on how to treat missing variables.

```
plot <- ggplot(data, aes(x=preis, fill = is.na(energieeffizienz)))
plot <- plot + geom_histogram(alpha = 0.6, binwidth=1000, colour="black")
plot + scale_fill_manual(values=c("red","blue"))
```



4.3 Preis - the dependent variable

Numerical distribution measures lead to observations:

- something
- something 2

```
kable(t(as.matrix(summary(data$preis))))
```

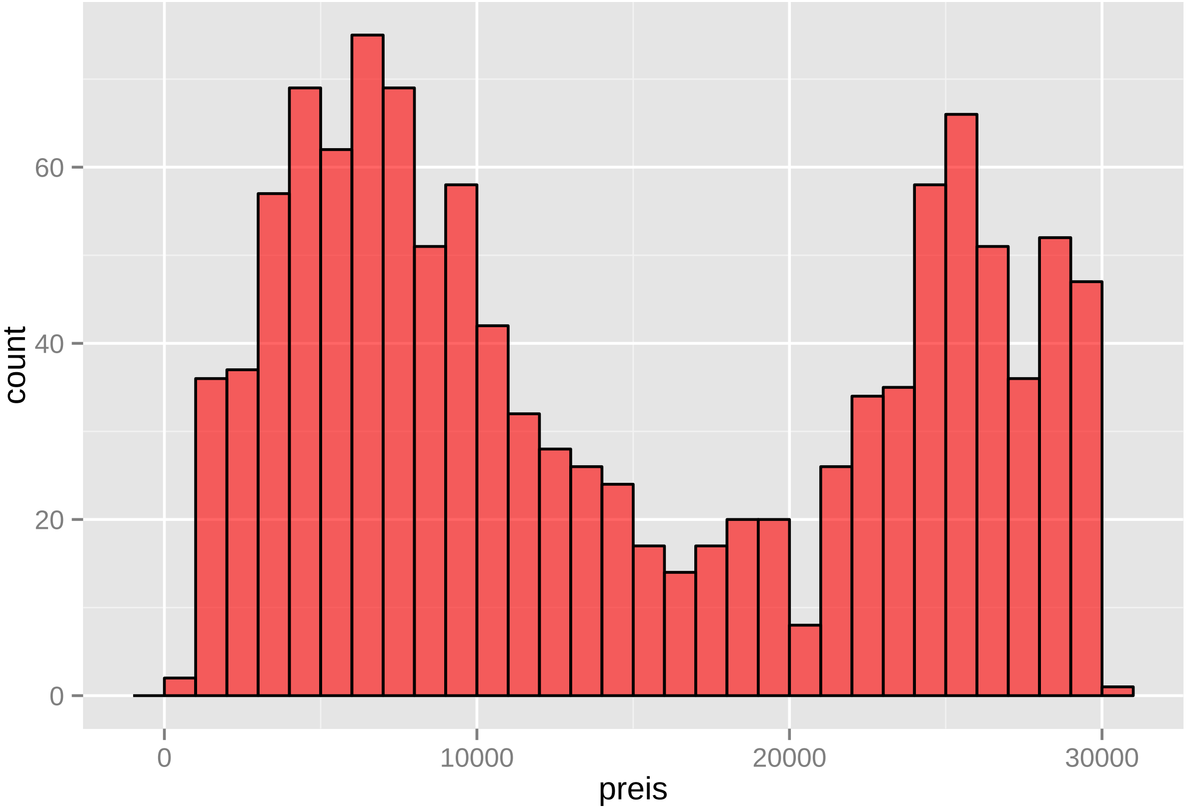
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
700	6800	11900	14790	24740	30000

Graphical representation by histogram. Obersations:

- Bimodal
- might be sampling problem

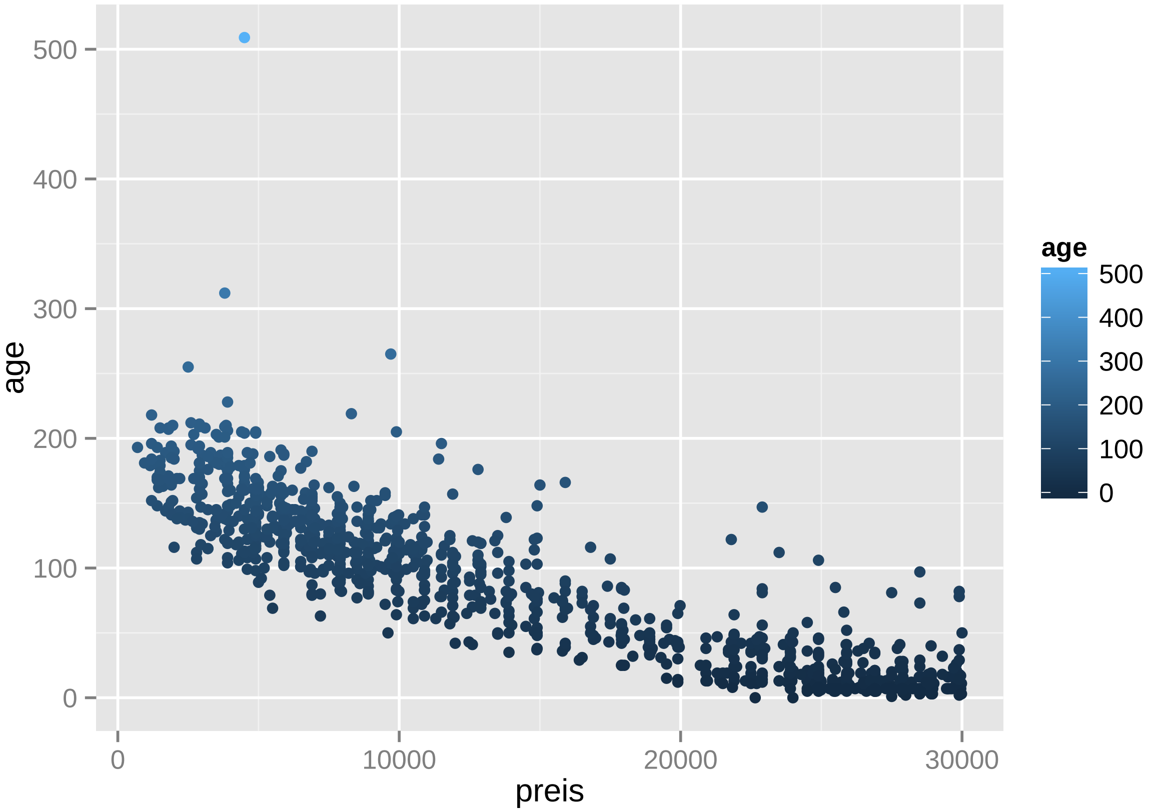
- might be other economic reasons in the composition of population

```
plot <- ggplot(data, aes(x=preis))
plot <- plot + geom_histogram(binwidth=1000, fill="red", alpha=0.6, color="black")
plot
```



‘Preis’ strongly dependent on ‘age’

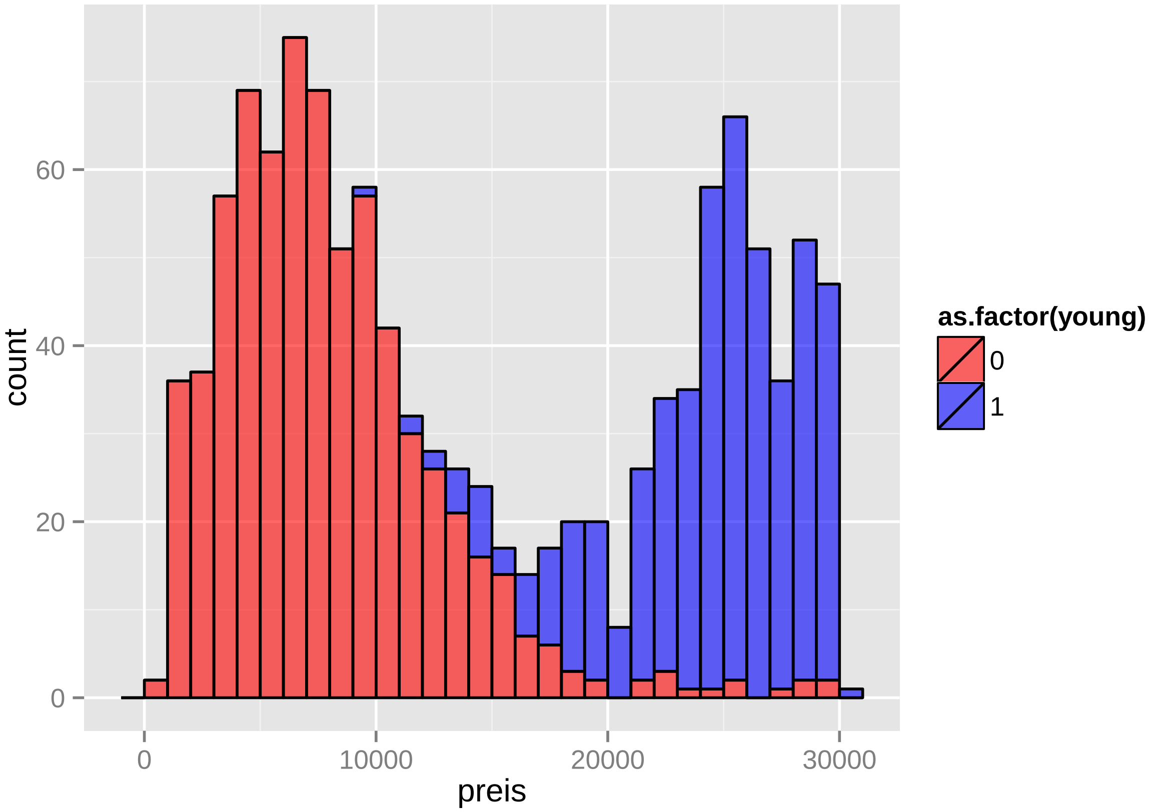
```
plot <- ggplot(data, aes(x=preis, y=age, colour = age)) + geom_point()
plot
```



Introducing variable ‘young’ (age $<$ 60 months) and differentiate histogram by ‘young’.

```
# add binary variable 'young' if age < 60 months (5years)
data$young <- ifelse(data$age <60, 1, 0)

# graphical distribution differenciated by 'young'
plot <- ggplot(data, aes(x=preis, fill=as.factor(young)))
plot <- plot + geom_histogram(binwidth=1000, alpha=0.6, colour="black")
plot + scale_fill_manual(values=c("red","blue"))
```



4.4 Numeric covariates of interest

Interested in mainly 2 aspects:

1. Dispersion of covariates
2. Correlation between covariates

Dispersion To get a better comparaility of dispersion some normalization or rescaling needed.

Decided to rescale all variables to the interval [0,1]. Where 0 is given by the minimum of the respective variable and 1 by the maximum.

Define function:

```
ReScaling <- function(x){
  x.min <- min(x, na.rm = TRUE)
  x.max <- max(x, na.rm = TRUE)
  y <- (x - x.min)/(x.max - x.min)
  return(y)
}
```

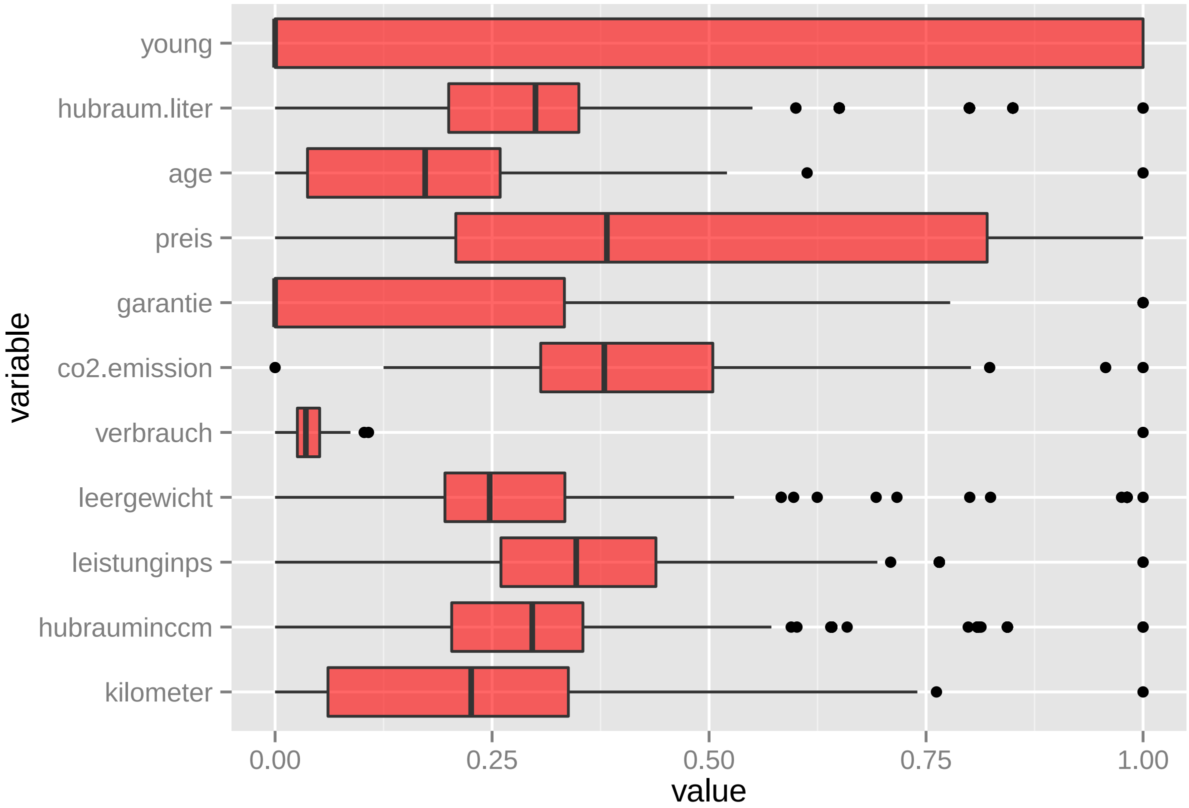
Rescale variables:

```
numeric.covariates <- names(data)[sapply(data, class) %in% c("integer","numeric")]

dataScaled <- data.frame(sapply(data[,numeric.covariates],FUN=ReScaling))
```

Plot Boxplots to illustrate dispersion:

```
dataScaledMelt <- melt(dataScaled)
plt <- ggplot(data=dataScaledMelt, aes(x=variable, y= value))
plt + geom_boxplot(fill="red", alpha=0.6) + coord_flip()
```



Correlation between covariates

```
# create correlation matrix
cor.matrix <- cor(data[,numeric.covariates], use="pairwise.complete.obs")
kable(cor.matrix[,1:5])
```

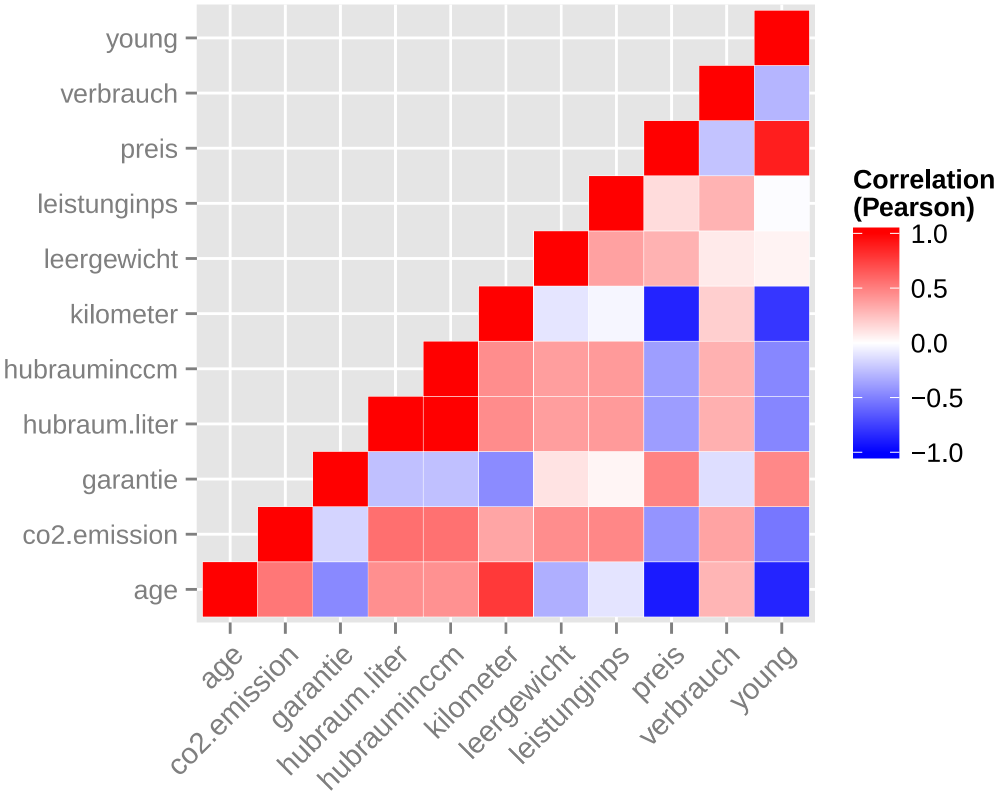
	kilometer	hubrauminccm	leistunginps	leergewicht	verbrauch
kilometer	1.0000000	0.4421278	-0.0326063	-0.1006008	0.1860555
hubrauminccm	0.4421278	1.0000000	0.3942003	0.3817734	0.3036200
leistunginps	-0.0326063	0.3942003	1.0000000	0.3680665	0.2953052
leergewicht	-0.1006008	0.3817734	0.3680665	1.0000000	0.0801746
verbrauch	0.1860555	0.3036200	0.2953052	0.0801746	1.0000000
co2.emission	0.3538358	0.5560042	0.4686437	0.4427186	0.3594363
garantie	-0.4530271	-0.2431581	0.0373400	0.1071301	-0.1297943
preis	-0.8605651	-0.3806537	0.1354277	0.2982627	-0.2302126
age	0.7749900	0.4261774	-0.1052117	-0.3100865	0.2883474
hubraum.liter	0.4464731	0.9994005	0.3948409	0.3792670	0.3082417
young	-0.7869761	-0.4670573	-0.0084273	0.0457022	-0.2882532

```r
kable(cor.matrix[,6:10])
```

	co2.emission	garantie	preis	age	hubraum.liter
kilometer	0.3538358	-0.4530271	-0.8605651	0.7749900	0.4464731
hubrauminccm	0.5560042	-0.2431581	-0.3806537	0.4261774	0.9994005
leistunginps	0.4686437	0.0373400	0.1354277	-0.1052117	0.3948409
leergewicht	0.4427186	0.1071301	0.2982627	-0.3100865	0.3792670
verbrauch	0.3594363	-0.1297943	-0.2302126	0.2883474	0.3082417
co2.emission	1.0000000	-0.1669743	-0.4144796	0.5315443	0.5621505
garantie	-0.1669743	1.0000000	0.4822365	-0.4597750	-0.2455172
preis	-0.4144796	0.4822365	1.0000000	-0.8962758	-0.3844113
age	0.5315443	-0.4597750	-0.8962758	1.0000000	0.4342286
hubraum.liter	0.5621505	-0.2455172	-0.3844113	0.4342286	1.0000000
young	-0.5324957	0.4654580	0.8805761	-0.8548750	-0.4717815

```r
source("./functions/corTiles.R")
```

```r
corTile(data[,numeric.covariates], use="pairwise.complete.obs")
```



4.5 Categorical variables

Selecting the relevant variables:

```r
# select relevant categorical variables (logical & factor)
categoric.variables <- c('fahrzeugart', 'aussenfarbe', 'getriebeart', 'antrieb', 'treibstoff', 'tueren'
kable(categoric.variables)
```

fahrzeugart
aussenfarbe
getriebeart
antrieb
treibstoff
tueren
sitze
zylinder
plattform
abmfk

Frequency tables

Categorical variables that allow for easy implementation of bianry dummy variables:

```r
kable(t(as.matrix(with(data, table(getriebeart)))))
```

Automat	Handschaltung
214	956

```r
kable(t(as.matrix(with(data, table(antrieb)))))
```

Allrad	Hinterradantrieb	Vorderradantrieb
174	4	987

```r
kable(t(as.matrix(with(data, table(treibstoff)))))
```

Benzin	Diesel	Gas (LNG, LPG, GPL, CNG)
776	388	5

```r
kable(t(as.matrix(with(data, table(abmfk)))))
```

FALSE	TRUE
652	518

Categorical variables that are highly concentrated on one value:

```r
kable(t(as.matrix(with(data, table(fahrzeugart)))))
```

Occasion	Oldtimer	Vorfuehrwagen
1146	1	21

```r
kable(t(as.matrix(with(data, table(tueren)))))
```

3	4	5
8	108	1052

```r
kable(t(as.matrix(with(data, table(sitze)))))
```

2	3	4	5	6	7	9
2	3	1	1068	1	37	1

```r
kable(t(as.matrix(with(data, table(zylinder)))))
```

4	5	6
1088	48	27

Categorical variables that are multidimensional:

```r
kable(t(as.matrix(with(data, table(aussenfarbe)))))
```

anthrazit	beige	blau	bordeaux	braun	gelb	gold	grau	gruen	orange	rot	schwarz	silber	violett	weiss
22	4	170	11	7	5	2	171	66	1	45	281	283	1	100

```r
kable(t(as.matrix(with(data, table(plattform)))))
```

Bora	Caddy	Golf	LT	Multivan	Passat	VAR	Variant
83	121	540	4	19	401	1	1