ICY0006 - Probability Theory and Statistics - Matryg project, TalTech

- Stage 1
  - Introduction
  - The work environment
  - The dataset
  - Variables
    - Missing data
    - Data visualization
    - Interpreting the visualization
- Stage 2
  - Quantitative overview of data
    - Central tendency measures
    - Variability measures
    - Analysis
- Stage 3
  - Linear relationships between variables
    - Removing outliers
    - Correlation matrix and visualization
  - Scatterplot - dependent variable vs. independent variable
  - Linear regression model
  - Regression with multiple variables
- Stage 5 - Training and testing regression model
  - Splitting the data set
  - Training the model
  - Testing the model
    - RMSE
    - MAE
    - MAPE
- Lottery - Stage 4
  - Viking Lotto

Stage 1

Introduction

This is my project for the course ICY0006 - Probability Theory and Statistics at TalTech University. Among the content of my project is a matryg_Notebook.Rmd-file and from that file a html-file can be generated and read as a report. Any questions related to the report can be sent to my mail: matryg@ttu.ee (mailto:matryg@ttu.ee)
The link to my GitHub repo is https://github.com/matry18/matrygV3 (https://github.com/matry18/matrygV3)
Enjoy the reading!

The work environment

For this project I have used RStudio as my IDE. I have used version control with Git and continuously committed my work to GitHub, where the repository is public accessible.
I have used the Notebook function in RStudio to generate the matryg_Notebook.Rmd-file from where a .html-file was generated and can be read as the report for this project.
The development of the report was an ongoing process following the progression of the course and continuous implementing the topics taught in the lecture and practices.

The dataset

The original dataset for this project was based on house prices and computer generated. This caused some issues since the dataset was not suitable due to a certain problem. Because of that the dataset was changed 4 weeks into the course. Doing the 8th week of the course I faced some problems with all the possible linear regressions in the dataset being heteroscedastic. This lead to another change of the dataset. Therefore this dataset is version 3 in my project.

This dataset is about lung capacity and is public available at https://www.kaggle.com/aakashmisra/lung-capacity (https://www.kaggle.com/aakashmisra/lung-capacity) with an unknown (public) license. The version of the dataset is version 1 and was created the 28/9 2020 by the user Aakash Misra - https://www.kaggle.com/aakashmisra (https://www.kaggle.com/aakashmisra)
The dataset is a .csv-file and I downloaded it and unzipped it before importing it. When doing the importing of the dataset I used a dynamic file.choose() method where I chose the downloaded and unzipped file.

```
data1 <- read.csv(file.choose(), header=TRUE)
View(data1)
attach(data1)
```

To be able to manipulate and visualize the data set the following libraries and dependencies was loaded.

```
##Data Manipulation
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0
--
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts()
--
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

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

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(forecast)
library(MLmetrics)
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following object is masked from 'package:psych':
##
##     AUC
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

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

```
library(caTools)
#Visualisation
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(ggmap)
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
##
## Attaching package: 'ggmap'
```

```
## The following object is masked from 'package:plotly':
##
##     wind
```

```
library(knitr)
library(dygraphs)
library(ggthemes)
library(reshape)
```

```
##
## Attaching package: 'reshape'
```

```
## The following object is masked from 'package:plotly':
##
##     rename
```

```
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
## The following object is masked from 'package:dplyr':
##
##     rename
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:reshape':
##
##     colsplit, melt, recast
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(ggplot2)
library(ggcorrplot)
library(ggstatsplot)
```

```
## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car
```

```
## In case you would like cite this package, cite it as:
##      Patil, I. (2018). ggstatsplot: "ggplot2" Based Plots with Statistical Deta
ils. CRAN.
##      Retrieved from https://cran.r-project.org/web/packages/ggstatsplot/index.h
tml
```

Variables

In the following I will describe the variables in the data set.
The data set consists of 725 observations and has 6 variables.

LungCap is short for lung capacity also know as total lung capacity (TLC). Lung capacity is the volume of air in the lungs upon the maximum effort of inspiration. Among healthy adults, the average lung capacity is about 6 liters. Age, gender, body composition, and ethnicity are factors affecting the different ranges of lung capacity among individuals. TLC rapid increases from birth to adolescence and plateaus at around 25 years old. Males tend to have a greater TLC than females, while individuals with tall stature tend to have greater TLC than those with short stature, and individuals with a high waist-to-hip ratio generally have a lower TLC. Individuals of African descent have a lower TLC compared to individuals of European descent. Additional factors that affect an individual's lung capacity include the level of physical activity, chest wall deformities, and respiratory diseases.[1]
The lung capacity is a dependent variable since it is influenced by age, gender etc. It is a continuous value as well as quantitative. It is measured on an interval scale.

Age indicates the persons age in years. This variable is independent, quantitative and discrete since all the age values are presented as integer values on an interval scale.

Height is the height in inches. This is an independent, continuous, quantitative variable measured on an interval scale. This variable must be transformed into a measurement of meter. Therefore I multiply the height value with 0.0254 to get the value in meter. This value has been added to the dataset as the variable **HeightMeter** and share the same characteristics as Height.

```
data1$HeightMeter <- data1[,3]*0.0254
View(data1)
```

The **Smoke** variable is independent, discrete and qualitative. It defines if the person is smoking, since it influences the lung capacity. The categorical values is “yes” if the person smokes and “no” if the person does not smoke. This is measured on a nominal scale. As an alternative the true/false-value could have been indicated with numbers, where 1 is true and 0 is false, but for now I will not manipulate the value.

Gender defines the gender of the person. This variable is independent, discrete and qualitative measured on a nominal scale. If the person is male the value is “male” and if the person is female the value is “female”. As an alternative the true/false-value could have been indicated with numbers, where 1 is true and 0 is false, but for now I will not manipulate the value.

The **Caesarean** indicates how the person was born. If the person was born by caesarean the value is “yes” and if not the value is “no”. This is an independent, discrete, qualitative variable measured on a nominal scale. As an alternative the true/false-value could have been indicated with numbers, where 1 is true and 0 is false, but for now I will not manipulate the value.

Missing data

Before going any further with the dataset I will check for any missing values by the use of the inbuilt function in RStudio.

```
which(is.na(data1))
```

```
## integer(0)
```

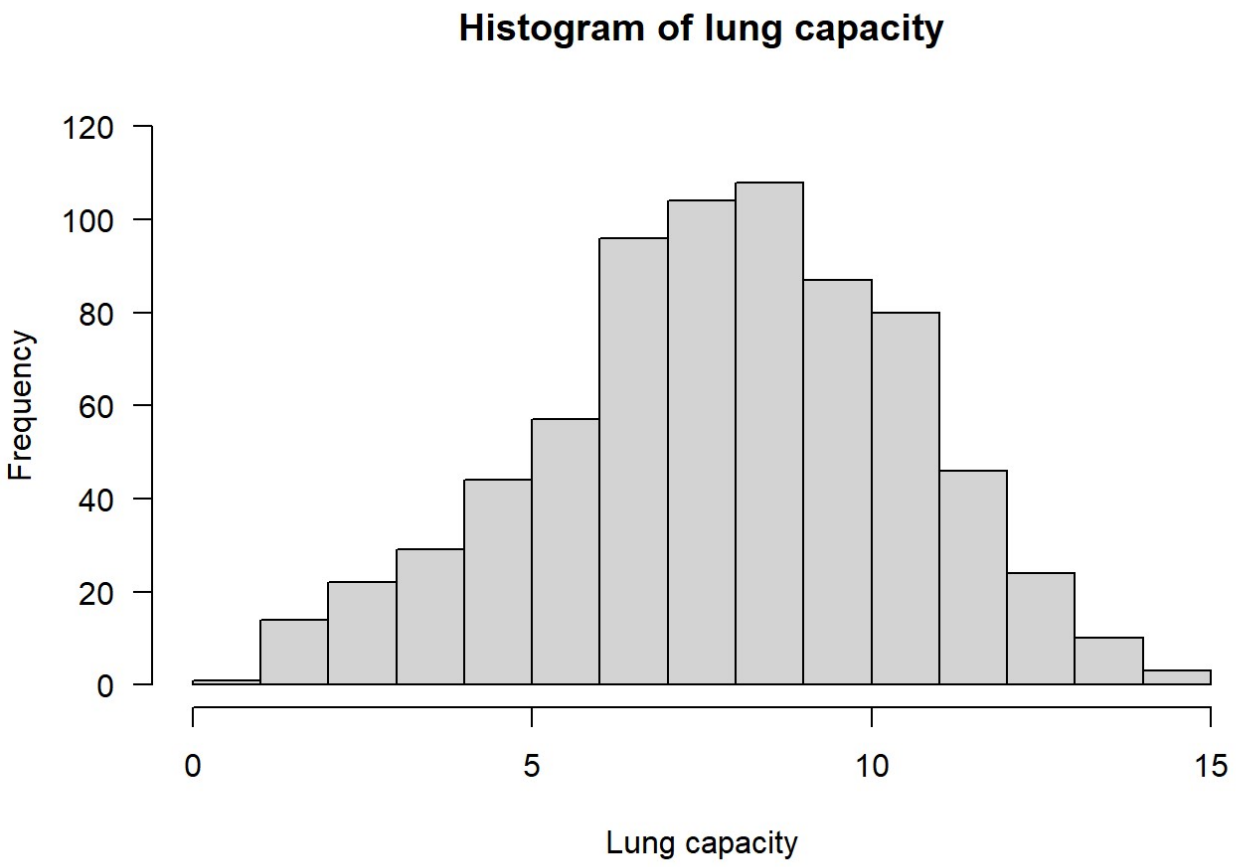
Since the returned value is zero that means that I don’t have any missing data in my dataset.

Data visualization

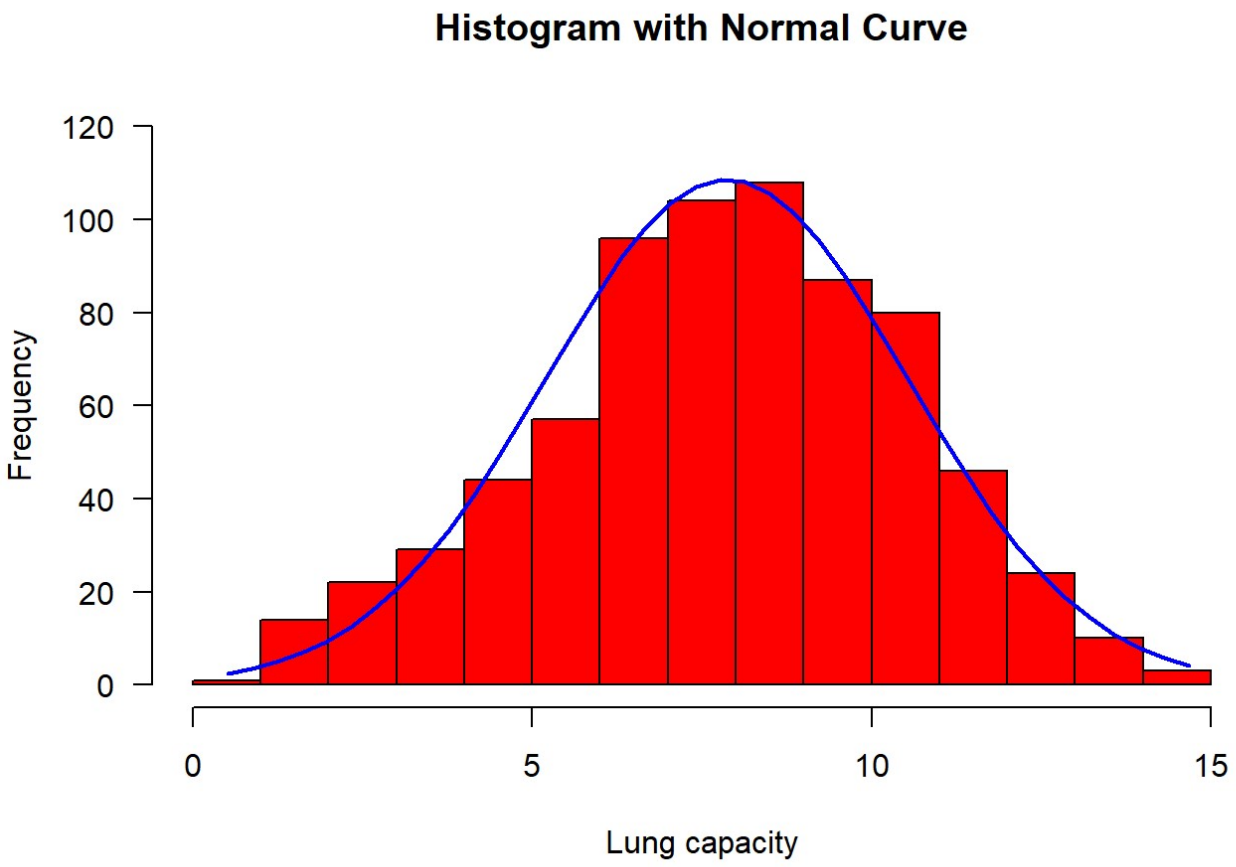
In the following I will visualize each variable. For Lung capacity, age and height in meters I have used histograms. Since histograms are strongly affected by the number of bins and therefore makes it difficult to determine the shape of the distribution I have added a normal curve to the histograms.
Another way of viewing the variables could be with the use of Kernel Density plots. Therefore I have chosen to visualize the lung capacity, age and height in meters with this curve.

Lung capacity

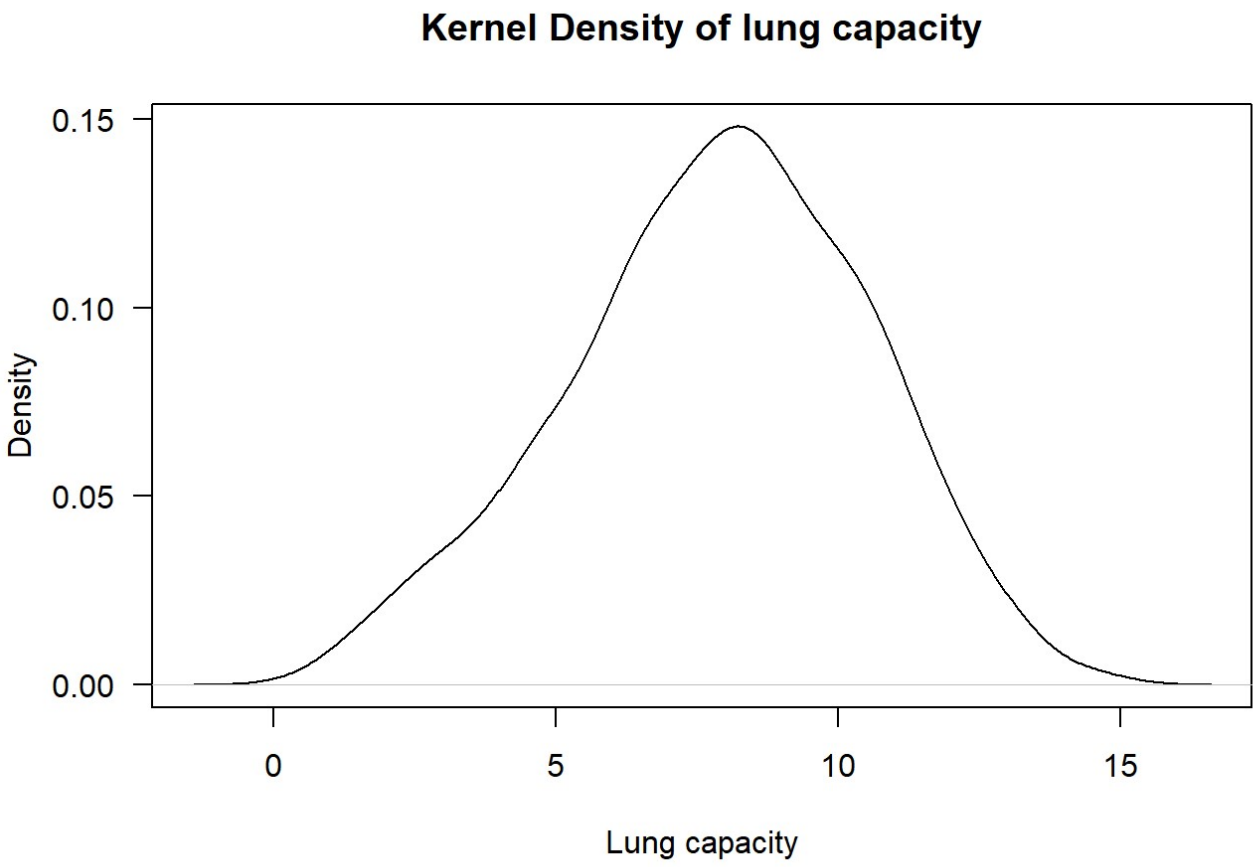
```
hist(data1$LungCap, las=1, ylim= c(0,120),xlab = "Lung capacity", main= "Histogram
of lung capacity")
```




```
x <- data1$LungCap
h<-hist(x, breaks=20, col="red", xlab="Lung capacity", las=1, main="Histogram with
Normal Curve", ylim = c(0,120), xlim = c(0,15))
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```

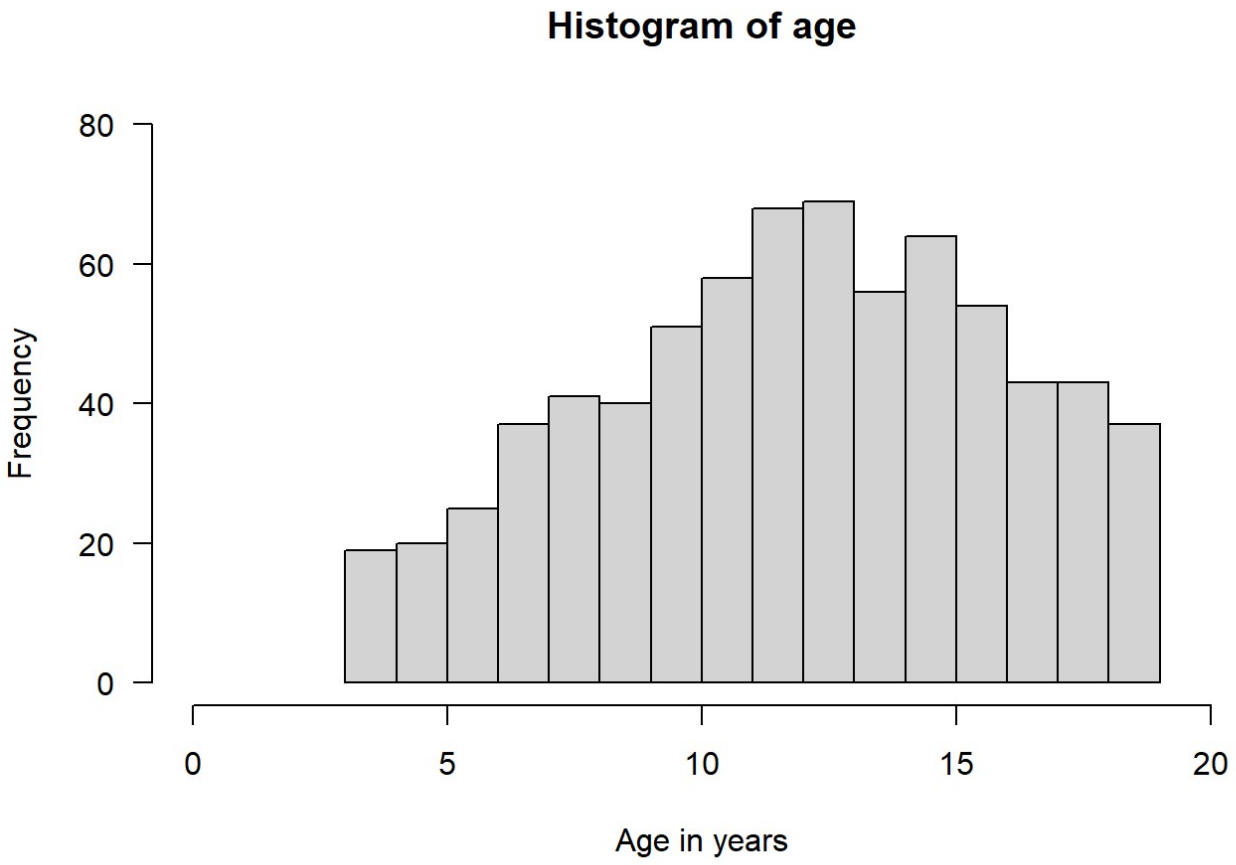



```
plot(density(data1$LungCap), main = "Kernel Density of lung capacity", xlab = "Lun
g capacity", las=1)
```

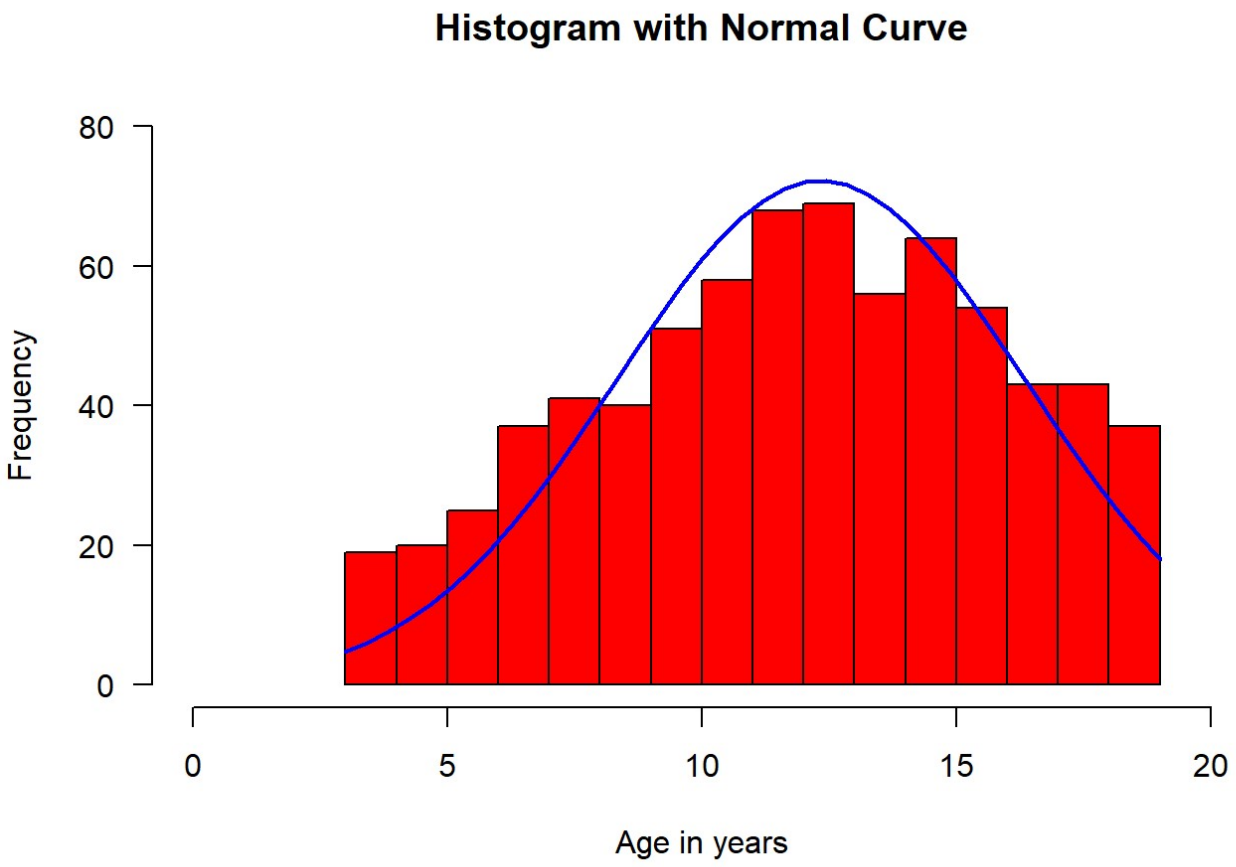


Age

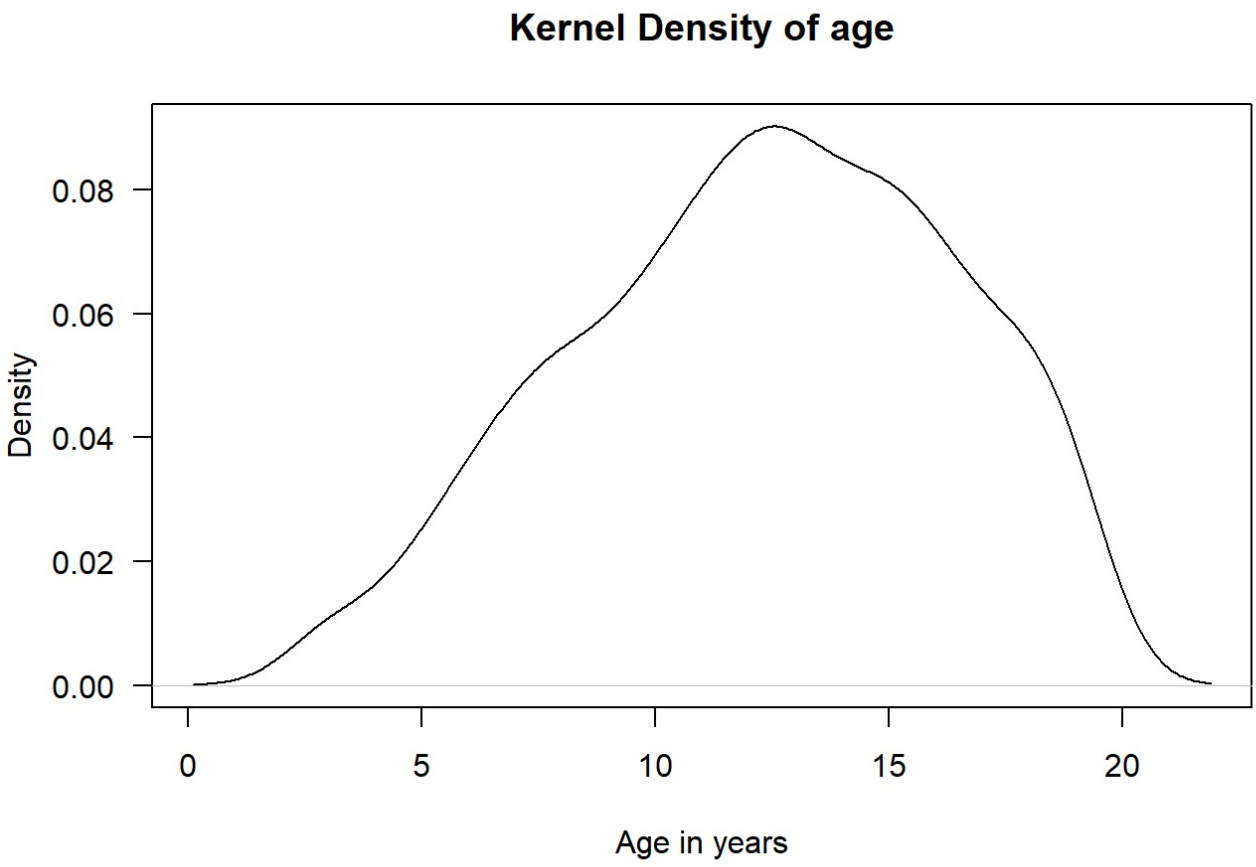
```
hist(data1$Age, xlim = c(0,20), ylim=c(0,80), las=1, xlab = "Age in years", main=
"Histogram of age", breaks = 20)
```




```
x <- data1$Age
h<-hist(x, breaks=20, col="red", xlab="Age in years", las=1, main="Histogram with
Normal Curve", ylim = c(0,80), xlim = c(0,20))
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```

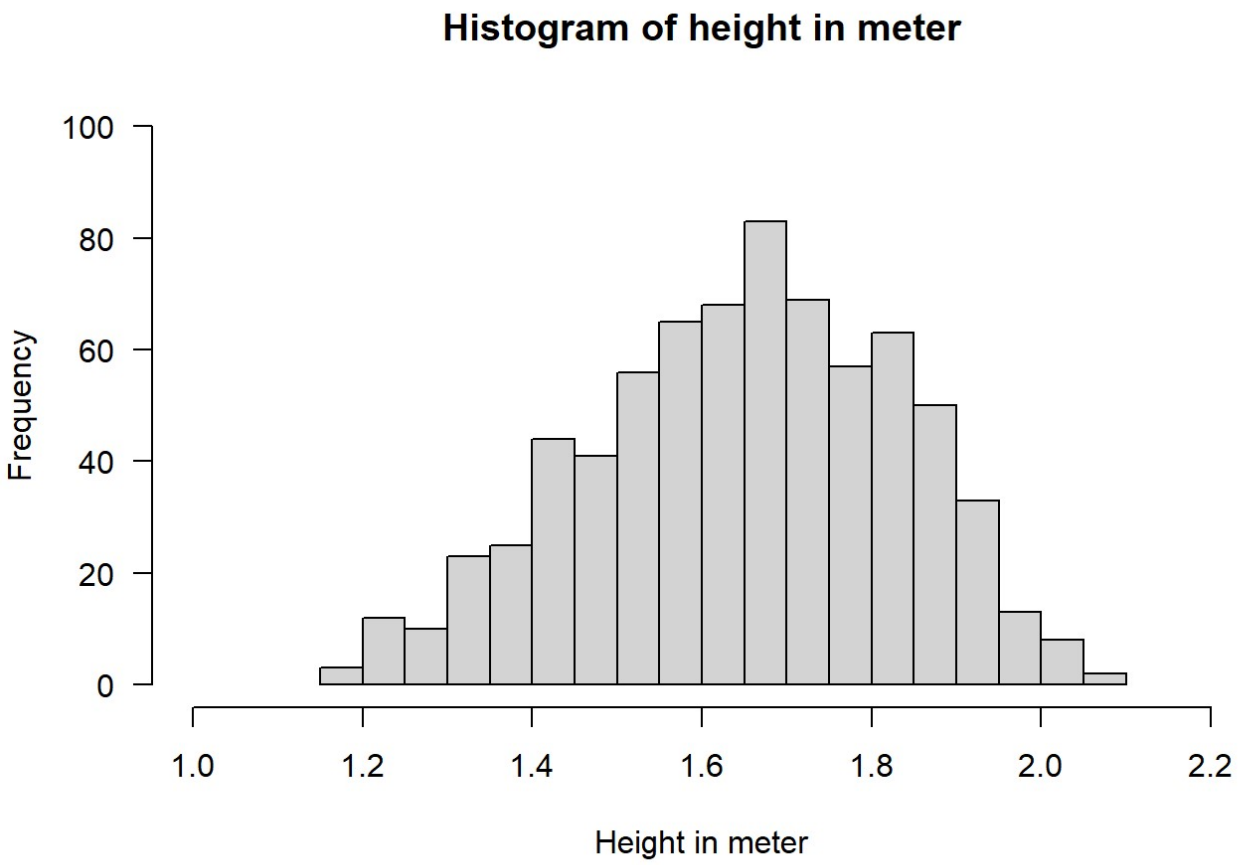



```
plot(density(data1$Age), main = "Kernel Density of age", xlab = "Age in years", la
s=1)
```

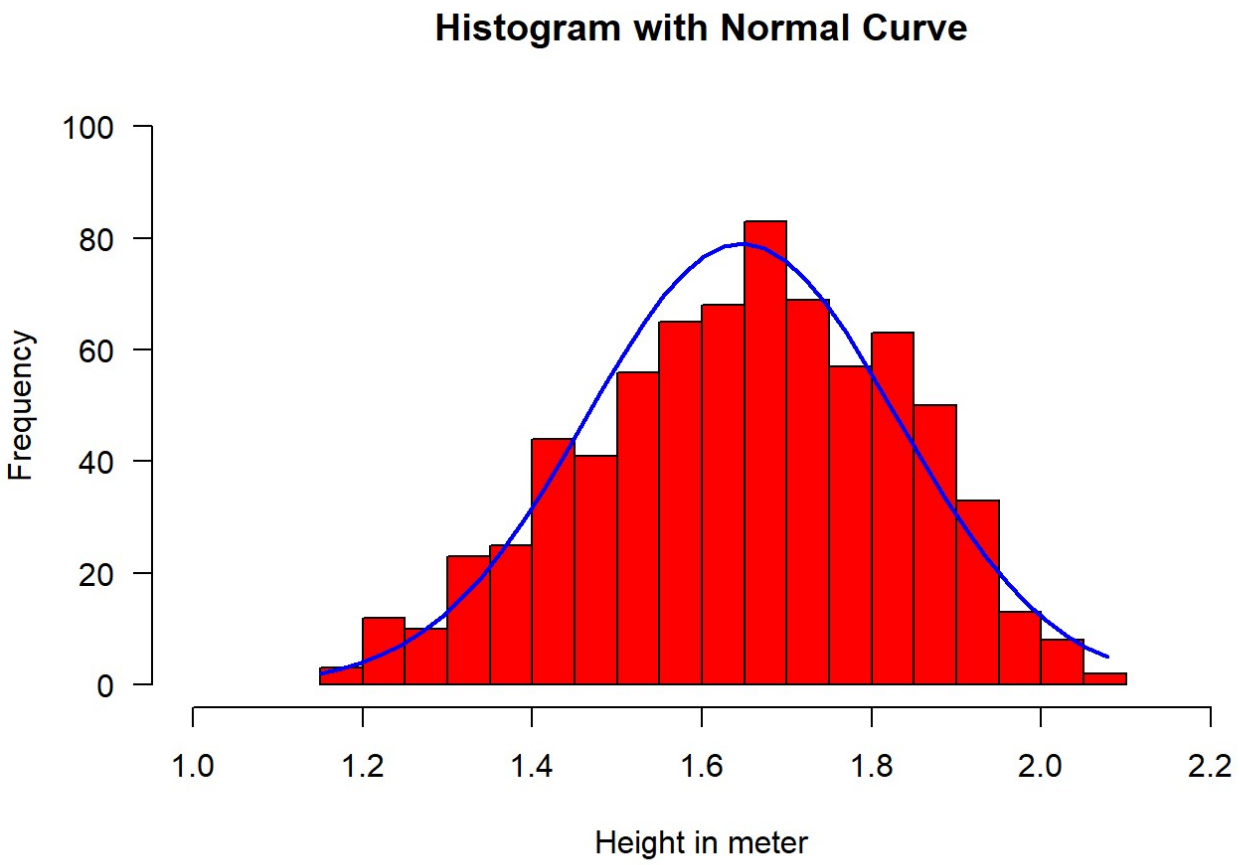


Height in meters

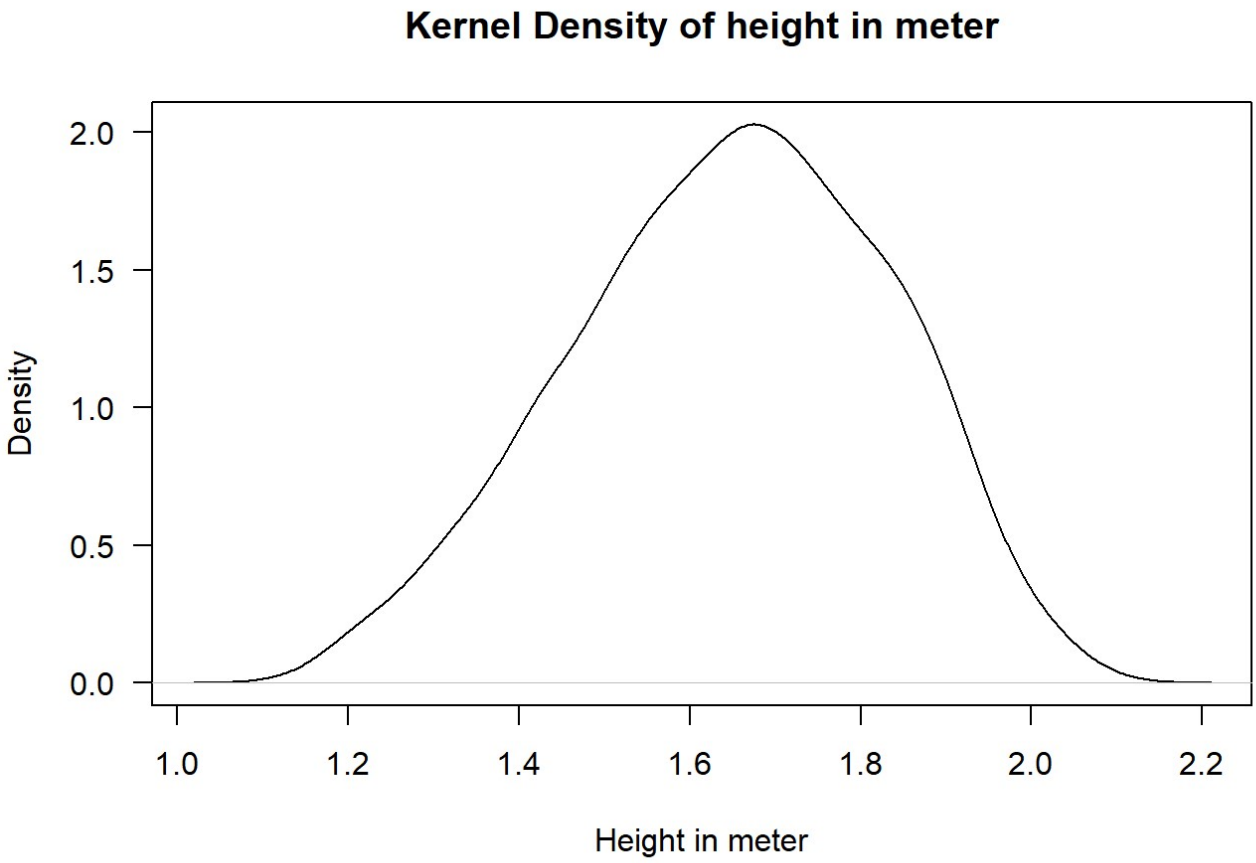
```
hist(data1$HeightMeter, xlim= c(1,2.2), ylim = c(0,100), las=1, xlab = "Height in
meter", main= "Histogram of height in meter", breaks = 20)
```




```
x <- data1$HeightMeter
h<-hist(x, breaks=20, col="red", xlab="Height in meter", las=1, main="Histogram wi
th Normal Curve", ylim = c(0,100), xlim = c(1,2.2))
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```



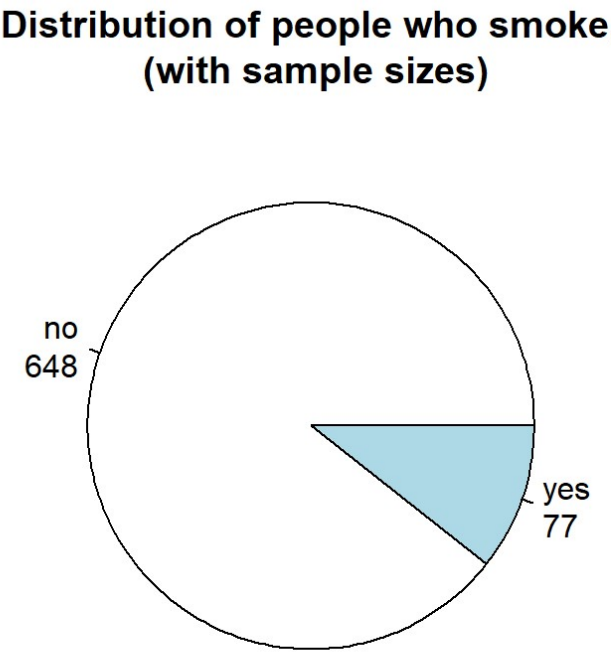

```
plot(density(data1$HeightMeter), main = "Kernel Density of height in meter", xlab
= "Height in meter", las=1)
```



In the following I will visualize the categorical variables with pie charts since it makes a good overview.

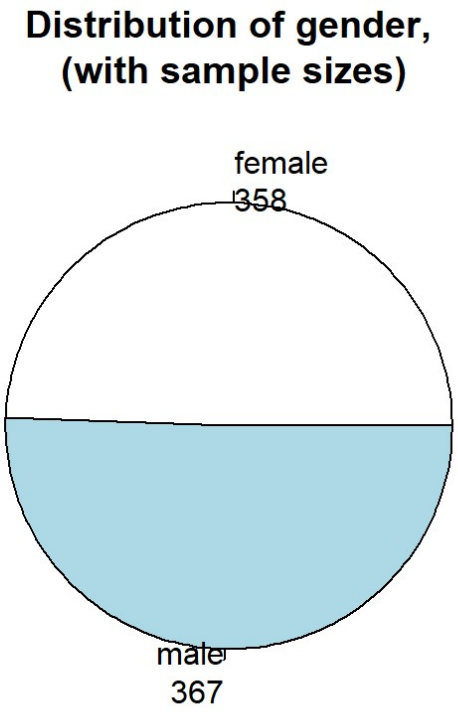
Smoke

```
smoke.count<-table(data1$Smoke)
lbls <- paste(names(smoke.count), "\n", smoke.count, sep="")
pie(smoke.count,border=T, labels = lbls,
   main="Distribution of people who smoke,\n (with sample sizes)")
```



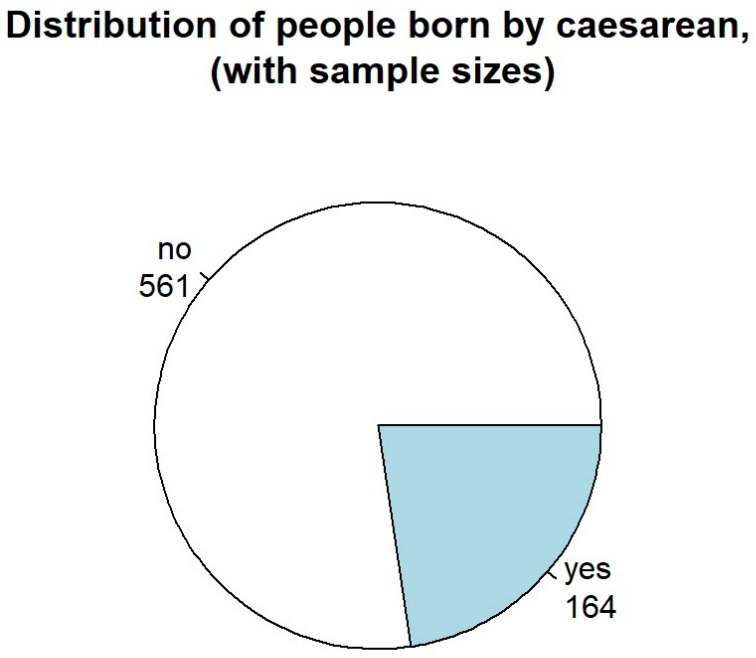
Gender

```
gender.count<-table(data1$Gender)
lbls <- paste(names(gender.count), "\n", gender.count, sep="")
pie(gender.count,border=T, labels = lbls,
   main="Distribution of gender,\n (with sample sizes)")
```



Caesarean

```
caesarean.count<-table(data1$Caesarean)
lbls <- paste(names(caesarean.count), "\n", caesarean.count, sep="")
pie(caesarean.count,border=T, labels = lbls,
   main="Distribution of people born by caesarean,\n (with sample sizes)")
```



Interpreting the visualization

In the following I will describe what can be seen from the graphs and charts.

The lung capacity has a nice bell shape, which indicates a normal distribution. The distribution is very close to be complete symmetrical with a single peek.
The age variable is close of having a nice bell shape but it tends to have a small negative skewness. It has a single peek and is close to being symmetrical but not complete symmetrical.
The height in meter has a very nice bell shape which indicates a normal distribution with a single peek and the shape of it is very close to be symmetrical.
The distribution of people who smoke indicates that approx. 1/8 of the people in the population do smoke and 7/8 does not.
The distribution of genders is close to be even with slightly more males than females in the population.
The distribution of people born by ceasarean shows that nearly 1/4 of the population is born by caesarean and 3/4 is not.

Stage 2

Quantitative overview of data

In the following I will compute the central tendency, also known as measure of central tendency or average. Basically this is the central value for a probability distribution. I will also compute the variability measures which is how “spread out” the data is. Then I will do a short analysis of the central tendencies and variability measures and compare the outcome with the visual representation of the distributions.

Central tendency measures

Mean, trimmed mean, median and mode

The mean is the sum of numerical values of each observation divided by the total number of observations, or put in a simpler way the sum of numbers divided by the number of numbers. The mean indicates at what point the distribution is in balance. In the following you will see the population mean for lung capacity, age and height in meters.

Lung capacity mean

```
lungCap.mean <- mean(data1$LungCap)
print(lungCap.mean)
```

```
## [1] 7.863148
```

Age mean

```
age.mean <- mean(data1$Age)
print(age.mean)
```

```
## [1] 12.3269
```

Since the age variable is discrete I will have to round off the mean value to not have a decimal number.

```
print(round(age.mean))
```

```
## [1] 12
```

Height in meters mean

```
heightMeter.mean <- mean(data1$HeightMeter)
print(heightMeter.mean)
```

```
## [1] 1.646841
```

With the trimmed mean I remove the lower and upper values in the distribution. In the following I will present a trimmed mean of 20 % for the same variables as above. This means that I will remove the lower 10 % and upper 10 % from the distribution and then recalculate the mean value.

Lung capacity trimmed mean of 20 %

```
mean(data1$LungCap, trim = 0.2)
```

```
## [1] 7.979425
```

Age trimmed mean of 20 %

```
mean(data1$Age, trim = 0.2)
```

```
## [1] 12.52874
```

And since age is a discrete variable it will have to be rounded off.

```
round(mean(data1$Age, trim = 0.2))
```

```
## [1] 13
```

Height in meter trimmed mean of 20 %

```
mean(data1$HeightMeter, trim = 0.2)
```

```
## [1] 1.654264
```

The median is the 50th percentile of the distribution which is the same as the value in the exact middle of the distribution. If the distribution is a even number, then I will take the average of the middle two values. The median can be found with the summary-function i R.

```
summary(data1)
```

```
##     LungCap            Age            Height         Smoke
##  Min.   : 0.507   Min.   : 3.00   Min.   :45.30   Length:725
##  1st Qu.: 6.150   1st Qu.: 9.00   1st Qu.:59.90   Class :character
##  Median : 8.000   Median :13.00   Median :65.40   Mode  :character
##  Mean   : 7.863   Mean   :12.33   Mean   :64.84
##  3rd Qu.: 9.800   3rd Qu.:15.00   3rd Qu.:70.30
##  Max.   :14.675   Max.   :19.00   Max.   :81.80
##     Gender           Caesarean          HeightMeter
##  Length:725         Length:725         Min.   :1.151
##  Class :character   Class :character   1st Qu.:1.521
##  Mode  :character   Mode  :character   Median :1.661
##                                        Mean   :1.647
##                                        3rd Qu.:1.786
##                                        Max.   :2.078
```

This shows the median for **lung capacity** is 8.0, for **age** it is 13.0 and for **height in meter** it is 1.661.

The mode is the most frequent value in the distribution for a given variable. To compute the mode I use a function.

```
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
```

Mode lung capacity

```
lungCap.mode <- getmode(data1$LungCap)
print(lungCap.mode)
```

```
## [1] 8.35
```

Mode age

```
age.mode <- getmode(data1$Age)
print(age.mode)
```

```
## [1] 13
```

Mode height in meter

```
heightMeter.mode <- getmode(data1$HeightMeter)
print(heightMeter.mode)
```

```
## [1] 1.66116
```

Variability measures

Range & interquartile range

The range of an observation variable is the difference of its largest and smallest data values. It is a measure of how far apart the entire data spreads in value.

Lung capacity range

```
print(range(data1$LungCap))
```

```
## [1]  0.507 14.675
```

Age range

```
print(range(data1$Age))
```

```
## [1]  3 19
```

Height in meter range

```
print(range(data1$HeightMeter))
```

```
## [1] 1.15062 2.07772
```

The interquartile range (IQR) of an observation variable is the difference of its upper and lower quartiles. It is a measure of how far apart the middle portion of data spreads in value. In other words it is the range of the middle 50 % of the scores in a distribution.

Lung capacity IQR

```
IQR(data1$LungCap)
```

```
## [1] 3.65
```

Age IQR

```
IQR(data1$Age)
```

```
## [1] 6
```

Height in meter IQR

```
IQR(data1$HeightMeter)
```

```
## [1] 0.26416
```

Variance & Standard Deviation

The variance is a numerical measure of how the data values is dispersed around the mean. Please notice that there is a difference in the variance depending on if it is from a population or sample! In the following I am looking at the variance on a population for lung capacity, age and height in meter.

Lung capacity variance

```
print(var(data1$LungCap))
```

```
## [1] 7.086288
```

Age variance

```
print(var(data1$Age))
```

```
## [1] 16.03802
```

The variance produces a measures of how far a set of numbers is spread out from their average value. Since age is a discrete value it has to be rounded off.

```
print(round(var(data1$Age)))
```

```
## [1] 16
```

Height in meter variance

```
print(var(data1$HeightMeter))
```

```
## [1] 0.03346502
```

The standard deviation (SD) of an observation variable is the square root of its variance. This is a useful measure of variability when the distribution is normal (or close to) because the proportion of the distribution within a given number of standard deviations from the mean can be calculated. Put in a more simple way, the SD can be used to measure the spread of the data in relation to the mean.

Lung capacity SD

```
print(sd(data1$LungCap))
```

```
## [1] 2.662008
```

Age SD

```
print(sd(data1$Age))
```

```
## [1] 4.00475
```

Again the value has to be an integer so it is rounded off.

```
print(round(sd(data1$Age)))
```

```
## [1] 4
```

Height in meter SD

```
print(sd(data1$HeightMeter))
```

```
## [1] 0.1829345
```

Analysis

Lung capacity

The mean and trimmed mean is very close to each other which indicates that no majority of data points is located in the outer 10 % of the lower or upper end of the distribution. The mean and the median is also very close to each other which indicates that the balance of the distribution is close to the center of the distribution. This would imply that the distribution is very close to a normal distribution which can be confirmed by the visual representation. The mode is 8.35 which corresponds very well with the histogram and curve since that is the peek. From the visual representation is can be difficult to read the exact value so the exact mode value is very helpful. The range values defines that start and end value for the variable, so the worst lung capacity is 0.507 and the best is 14.675. The IQR of 3.65 indicates that the middle 50 % of the data is not that wide spread in the distribution. The SD value (2.662008) is quite small which tells me that the spread of the data points from the mean is quite narrow. The variability measures corresponds very well with the graphical representation of the distribution.

Age

The mean and trimmed mean is actually really close to each other but when the value is rounded off the difference increases but the two values (12 and 13) is still close to each other. From this I can deduce that there is not many data points in the outer 10 % of the lower and upper part of the distribution. The mean and median is also very close to each other so the balance of the distribution is in the center of the distribution. The mode value is equal to the trimmed mean and median which indicates a normal distribution which can be confirmed by the visual representation of the histogram and density curve. Since the mean value is smaller than the median value a small negative skew can be seen. The central tendency values and the visual representation corresponds very well with each other. The youngest

person in the population is 3 years old and the oldest is 19 years old. The IQR value of 6 indicates that the central 50 % of the data is spread out among those 6 years. The SD value of 4 indicate how far from the mean value that the data is spread out. The value indicates that there is some spread among the observations for this variable but nothing of big concern. The variability measures correspons very well with the visual representation of the distribution.

Height in meter

The mean and trimmed mean is almost identical so no crucial data points is located in the outer 10 % of the scale for this variable. The same goes for the mode value which is identical with the median value. This indicates a symmetric distribution and when looking at the visual representation of the distribution this corresponds very well with the values. The shortest person in the observations is 1.15062 meter and the tallest person is 2.07772 meter. The IQR value of 0.26416 indicates that the central 50 % of the data is not that wide spread out. The SD value (0.1829345) is relatively small and indicates that the data for this variable is not that wide spread out. The variability measures corresponds very well with the graphical representation of the distribution.

I have chosen to leave out the categorical variables since it did not add any value to the project to analyze these.

In general the graphical representation gives the same impression as the values of central tendency measures and variability measures. However it can be difficult to read the exact values from the graphical representation and the central tendency measures and variability measures helps to understand the exact values.
When the values are rounded some details becomes hidden from the central tendency measures. The data representation can either be in numeric form or visual form depending on how you want to present the data and with what purpose. Usually it is easier to get an overview from the visual representation but the details is easier seen in the numerical representation of the data.

Stage 3

Linear relationships between variables

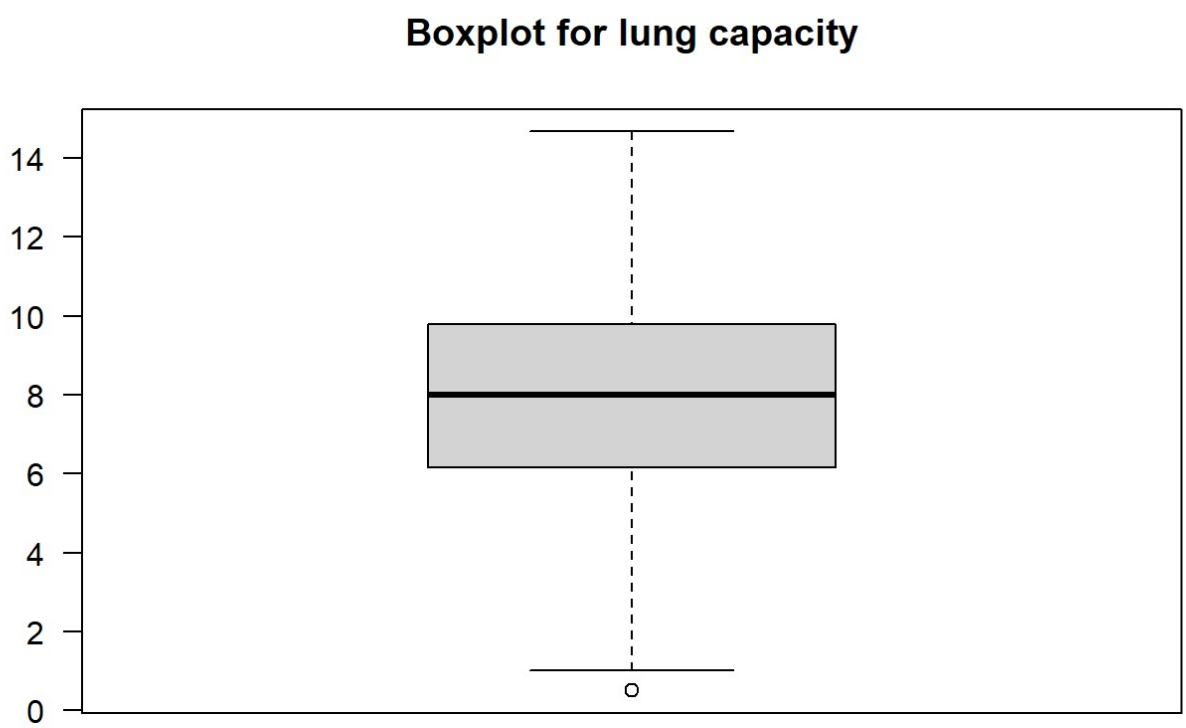
In the following I will select the suitable variables for linear relationships and remove outliers. Then I will compute a correlation matrix and visualize it with a heat map and explain what it shows.

As for variables I have selected lung capacity, age and height in meters since they are not categorical.

Removing outliers

Before I proceed with the linear relationship I will look into potential outliers. Often an outlier is an anomaly in the dataset that occurs because of the measurement errors, but also because the experiment being observed experiences momentary but drastic turbulence. The outliers can have a huge impact on the result of a linear regression model. The statistical parameters e.g. mean, standard deviation (SD) and correlation are impacted by outliers.[2] To locate the outliers I will use the IQR as it does not depend on the mean and SD.
The IQR is the central 50 % which is the same as the area above the 25th percentile of a distribution and under the 75th percentile of an distribution. A point is defined as an outlier if it is below the 25th or above the 75th percentile by a factor of 1,5 times the IQR. The outliers can be visualized in a boxplot, since the boxplot shows the median and the 1st and 3rd quartiles.
The below boxplot is just an example on how to visualize the outliers.

```
boxplot(data1$LungCap, las=1, main= "Boxplot for lung capacity")
```



From the above boxplot one outlier can be seen. but it can be difficult to read the exact value of the outlier in a boxplot. Therefore a statistical method for finding outliers is needed. By using the quantile() function I can find find the 25th and 75th percentile, and by using the IQR() function I can see the difference between the 75th and 25th percentiles.

```
# Lung capacity
Q_lung <- quantile(data1$LungCap, probs = c(0.25, 0.75), na.rm = FALSE)
iqr_lung <- IQR(data1$LungCap)
# Age
Q_age <- quantile(data1$Age, probs = c(0.25, 0.75), na.rm = FALSE)
iqr_age <- IQR(data1$Age)
# Height in meter
Q_heightMeter <- quantile(data1$HeightMeter, probs = c(0.25, 0.75), na.rm = FALSE)
iqr_heightMeter <- IQR(data1$HeightMeter)
```

From the above values I can find the cut-off ranges beyond which all data points are outliers.

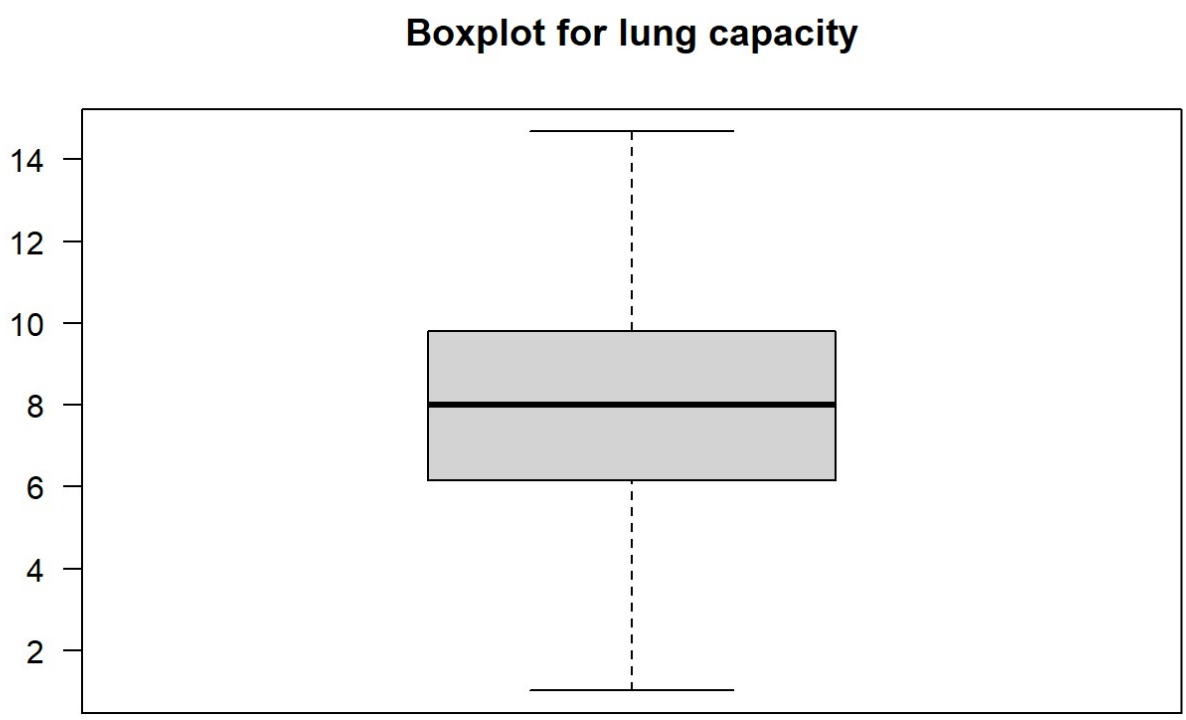
```
# Lung capacity
up_lung <-  Q_lung[2]+1.5*iqr_lung # Upper Range
low_lung<- Q_lung[1]-1.5*iqr_lung # Lower Range

# Age
up_age <-  Q_age[2]+1.5*iqr_age # Upper Range
low_age<- Q_age[1]-1.5*iqr_age # Lower Range

# Height in meter
up_heightMeter <-  Q_heightMeter[2]+1.5*iqr_heightMeter # Upper Range
low_heightMeter<- Q_heightMeter[1]-1.5*iqr_heightMeter # Lower Range
```

By using the subset() function I can extract the data between the lower and upper, leaving out the outliers.

```
data2=subset(data1, LungCap > low_lung & LungCap < up_lung & Age > low_age & Age <
up_age & HeightMeter > low_heightMeter & HeightMeter < up_heightMeter)
boxplot(data2$LungCap, las=1, main= "Boxplot for lung capacity")
```



When looking at the updated dataset (data2) it has 1 less observation than the old dataset (data1) which means that I only had 1 outlier.

```
str(data2)
```

```
## 'data.frame':    724 obs. of  7 variables:
##  $ LungCap    : num  6.47 10.12 9.55 11.12 4.8 ...
##  $ Age        : int  6 18 16 14 5 11 8 11 15 11 ...
##  $ Height     : num  62.1 74.7 69.7 71 56.9 58.7 63.3 70.4 70.5 59.2 ...
##  $ Smoke      : chr  "no" "yes" "no" "no" ...
##  $ Gender     : chr  "male" "female" "female" "male" ...
##  $ Caesarean  : chr  "no" "no" "yes" "no" ...
##  $ HeightMeter: num  1.58 1.9 1.77 1.8 1.45 ...
```

Correlation matrix and visualization

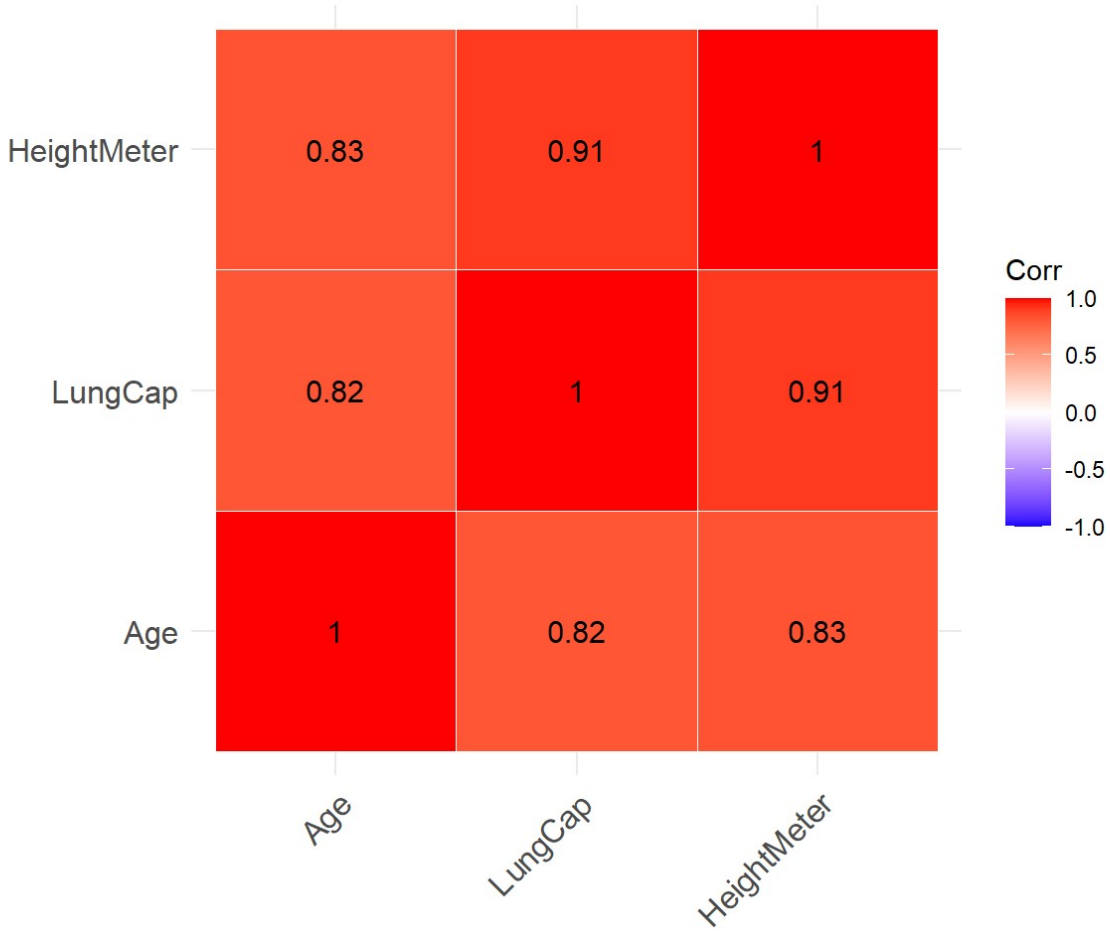
In the following I will compute a correlation matrix and visualize it with a heat map.

A heat map relates the numeric value from -1 to 1 into a color code where red represents a value of 1 and blue represents a value of -1. 0 indicates a non-correlation. To make the heat map a bit easier to read I have rounded the numerical value to two decimal numbers.

```
numVar <- data2 %>% select(LungCap, Age, HeightMeter)
cor(numVar)
```

```
##                LungCap       Age HeightMeter
## LungCap     1.0000000 0.8181640   0.9121143
## Age         0.8181640 1.0000000   0.8348995
## HeightMeter 0.9121143 0.8348995   1.0000000
```

```
corMatrix <- round(cor(numVar),2)
p.mat<-cor_pmat(numVar)
ggcorrplot(corMatrix,outline.col = "white",  hc.order = TRUE,type = "full",lab = T
RUE,p.mat = p.mat)
```



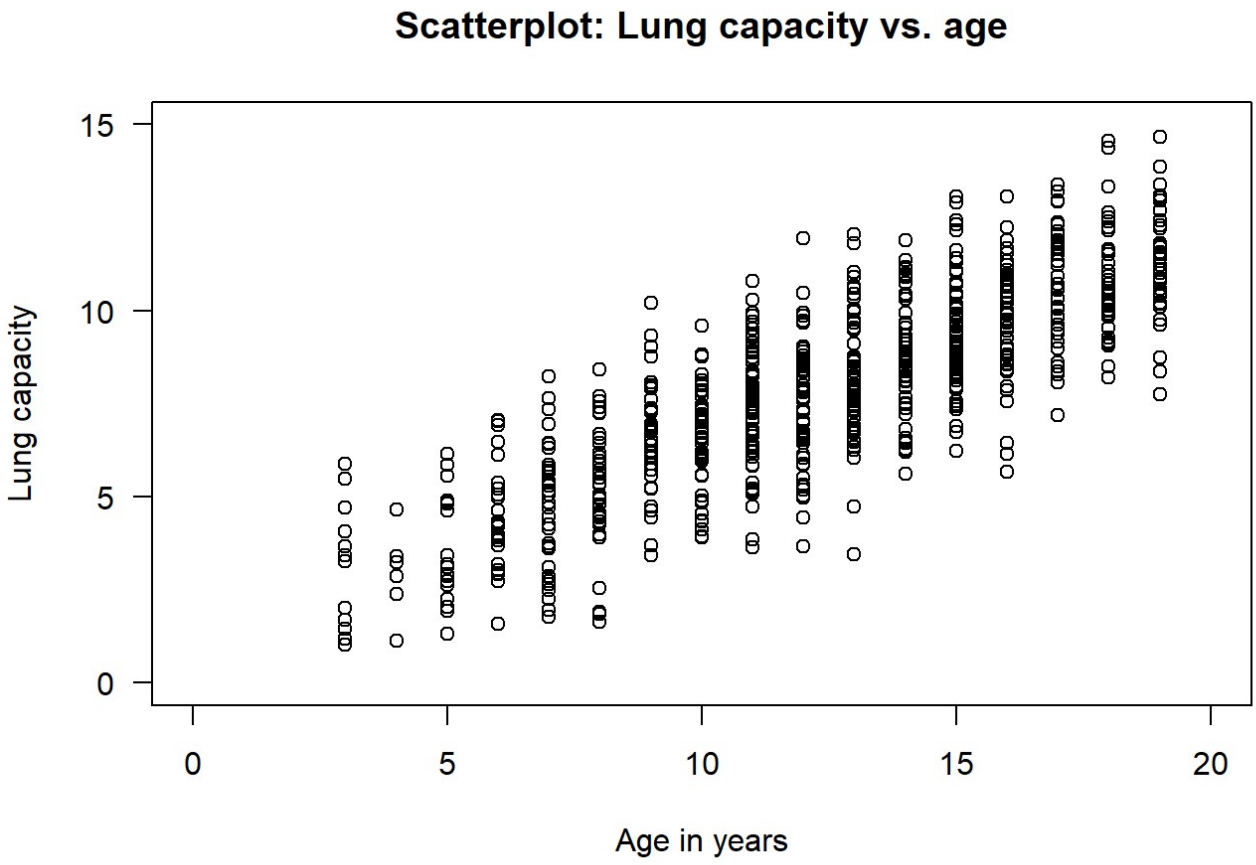
From the correlation matrix and the heat map there seems to be a very strong correlation between lung capacity and height in meter since the Pearson's correlation coefficient is 0.91. There is a strong correlation between age and lung capacity (0.82 Pearson's correlation coefficient) as well as between age and height in meter (0.83 Pearson's correlation coefficient). There is a perfect correlation between a variable and itself but that should be excluded and not be looked into.
The lung capacity must be the dependent variable, depending on both age and height in meter.

Scatterplot - dependent variable vs. independent variable

From the above correlation matrix I know that lung capacity is the dependent variable, where age and height in meter is the independent variables. In the following I will create scatterplots for respectively lung capacity vs. age, and lung capacity vs. height in meter.

Lung capacity vs. age

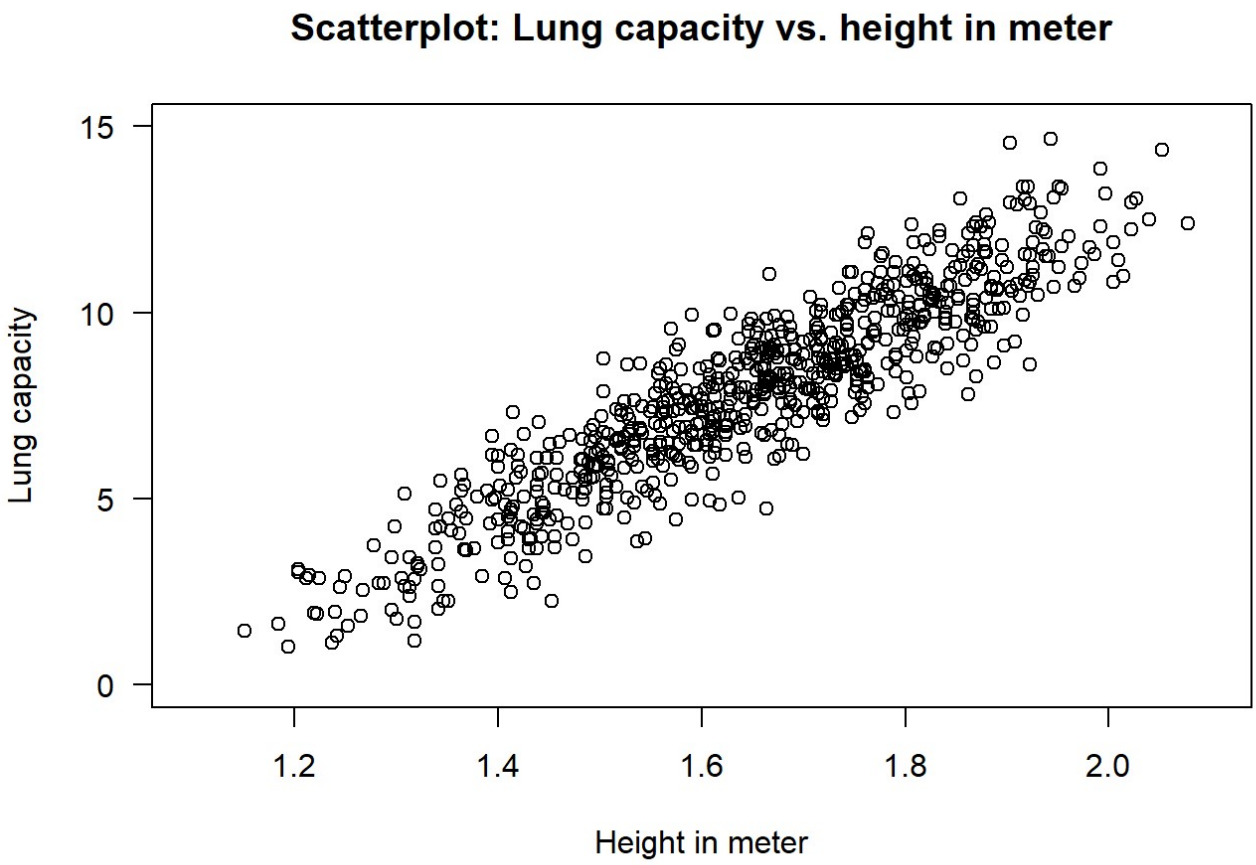
```
plot(data2$Age, data2$LungCap, main ="Scatterplot: Lung capacity vs. age", ylab =
"Lung capacity", xlab = "Age in years", las=1, xlim=c(0,20), ylim=c(0,15))
```



From the scatterplot I can see a strong relationship between age and lung capacity. The higher the age value the higher the lung capacity value. The two variables has a positive association. I can also observe a linear relationship since the points clusters along a straight (wide) line. The scatterplot shows homoscedasticity since all the random variables have the same finite variance. From the scatterplot it is visible that the age variable is discrete since all the points forms a vertical “line”.

Lung capacity vs. height in meter

```
plot(data2$HeightMeter, data2$LungCap, main ="Scatterplot: Lung capacity vs. heigh
t in meter", ylab = "Lung capacity", xlab = "Height in meter", las=1, xlim=c(1.1,
2.1), ylim=c(0,15))
```



The scatterplots shows a strong relationship between lung capacity and height in meter. It is easy to see that when the value for height in meter increases the lung capacity increases as well. I can see a positive linear relationship between the two variables. The scatterplot is in a homoscedastic form. I can see that both variables are continuous since the points in the plot is not in a vertical or horizontal “line”.

Linear regression model

In the following I will look into a linear regression model and visualize it with a scatter plot with a line. I have chosen lung capacity vs. height in meter, since it has the highest correlation coefficient value.

```
cor(data2$LungCap,data2$HeightMeter)
```

```
## [1] 0.9121143
```

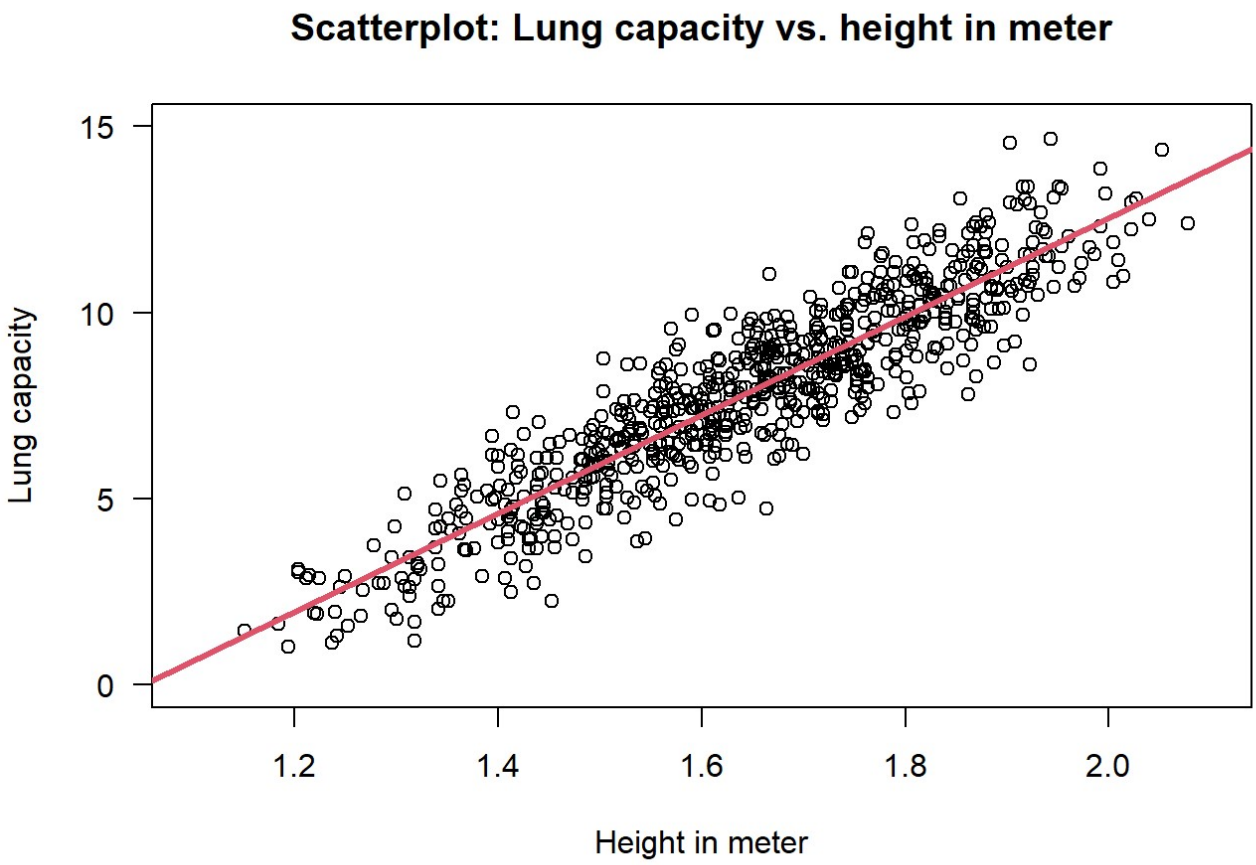
```
mod <- lm(data2$LungCap ~ data2$HeightMeter)
summary(mod)
```

```
##
## Call:
## lm(formula = data2$LungCap ~ data2$HeightMeter)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3653 -0.7001 -0.0060  0.7681  3.3001
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -13.9263     0.3669  -37.96   <2e-16 ***
## data2$HeightMeter  13.2335     0.2213   59.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.087 on 722 degrees of freedom
## Multiple R-squared:  0.832,  Adjusted R-squared:  0.8317
## F-statistic:  3574 on 1 and 722 DF,  p-value: < 2.2e-16
```

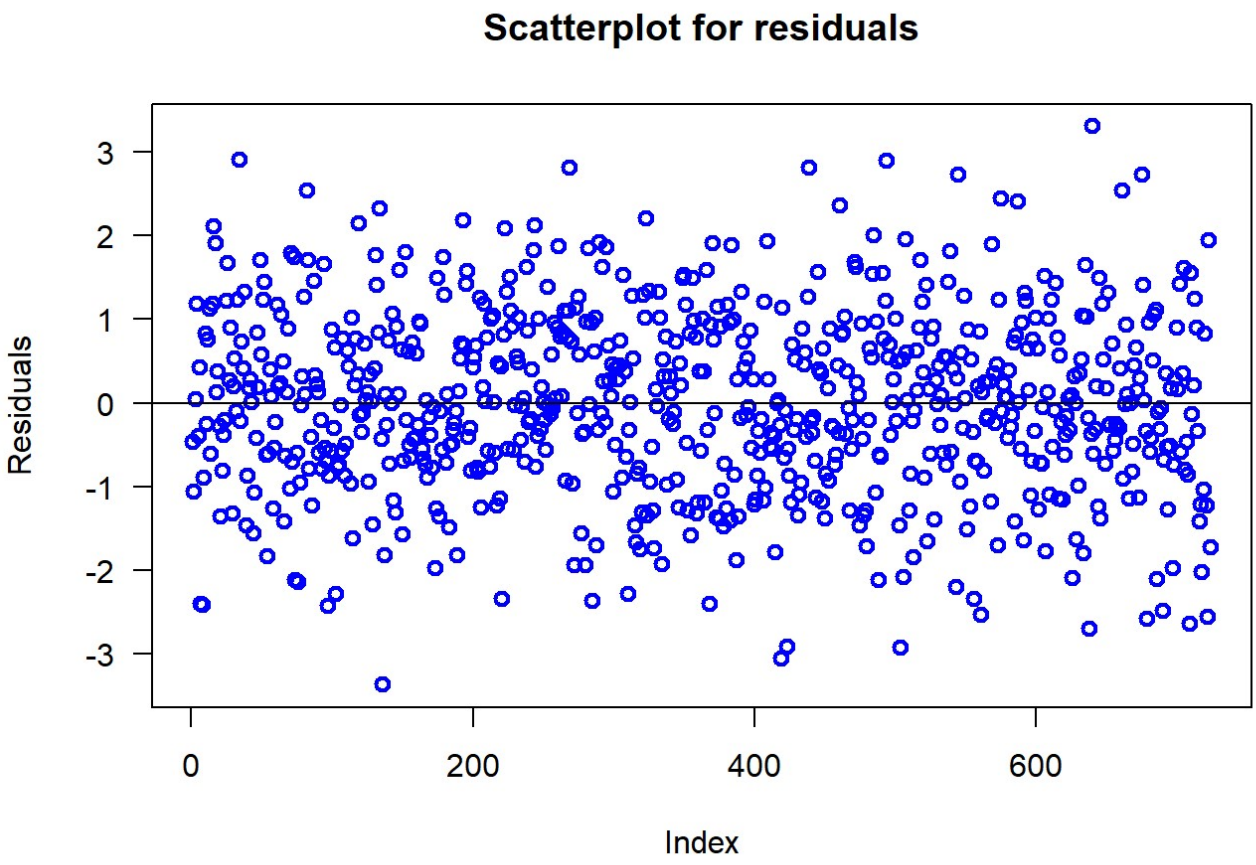
```
mod$coefficients
```

```
##       (Intercept) data2$HeightMeter
##         -13.92629          13.23349
```

```
plot(data2$HeightMeter, data2$LungCap, main ="Scatterplot: Lung capacity vs. heigh
t in meter", ylab = "Lung capacity", xlab = "Height in meter", las=1, xlim=c(1.1,
2.1), ylim=c(0,15))
abline(mod, col=2, lwd=3)
```




```
plot(mod$residuals, col="blue", lwd=2, ylab="Residuals", las=1, main = "Scatterplo
t for residuals")
abline(h=0)
```



In the summary the residuals (errors) are shown, e.g. the residual value for median is -0.0060. The estimate for the intercept is -13.9263 and it's standard error is 0.3669. The regression coefficient is the average change in the dependent variable (lung capacity) for a 1-unit change in the independent variable (height in meter). It is the slope of the regression line. For height in meter the estimate is 13.2335 and the standard error is 0.2213. The residual standard error is 1.087 which is a measure of a variation of observations around the regression line. This is the same as the square root of the mean squared error. The residual standard error tells how far observed lung capacity are from the predicted or fitted lung capacity. The multiple R-squared values (0.832) means that approx. 83 % of variation of lung capacity can be explained by this model.
I can see that the height in meter is the cause of the lung capacity. Both of the variables is continuous and have a strong positive linear relationship. The red line (the least squared regression line) is the prediction of the lung capacity based on the height in meter. All the points are the actual data values. The distance from a point to the line is the error called residual, since it is the difference between the actual data point and the predicted value. The line passes through the mean of both variables (lung capacity and height in meter).
The scatterplot for the residuals shows a random pattern.

Regression with multiple variables

In the following I will look into regression with multiple variables. I will use lung capacity as the dependent variable and age and height in meter as the independent variables to fit my linear regression model.

```
model1 <- lm(data2$LungCap ~ data2$Age + data2$HeightMeter)
summary(model1)
```

```
##
## Call:
## lm(formula = data2$LungCap ~ data2$Age + data2$HeightMeter)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4101 -0.7042 -0.0092  0.7111  3.1756
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -11.72674    0.47545 -24.665  < 2e-16 ***
## data2$Age           0.12409    0.01782   6.964 7.48e-12 ***
## data2$HeightMeter  10.96872    0.38955  28.158  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.053 on 721 degrees of freedom
## Multiple R-squared:  0.8425, Adjusted R-squared:  0.8421
## F-statistic:  1929 on 2 and 721 DF,  p-value: < 2.2e-16
```

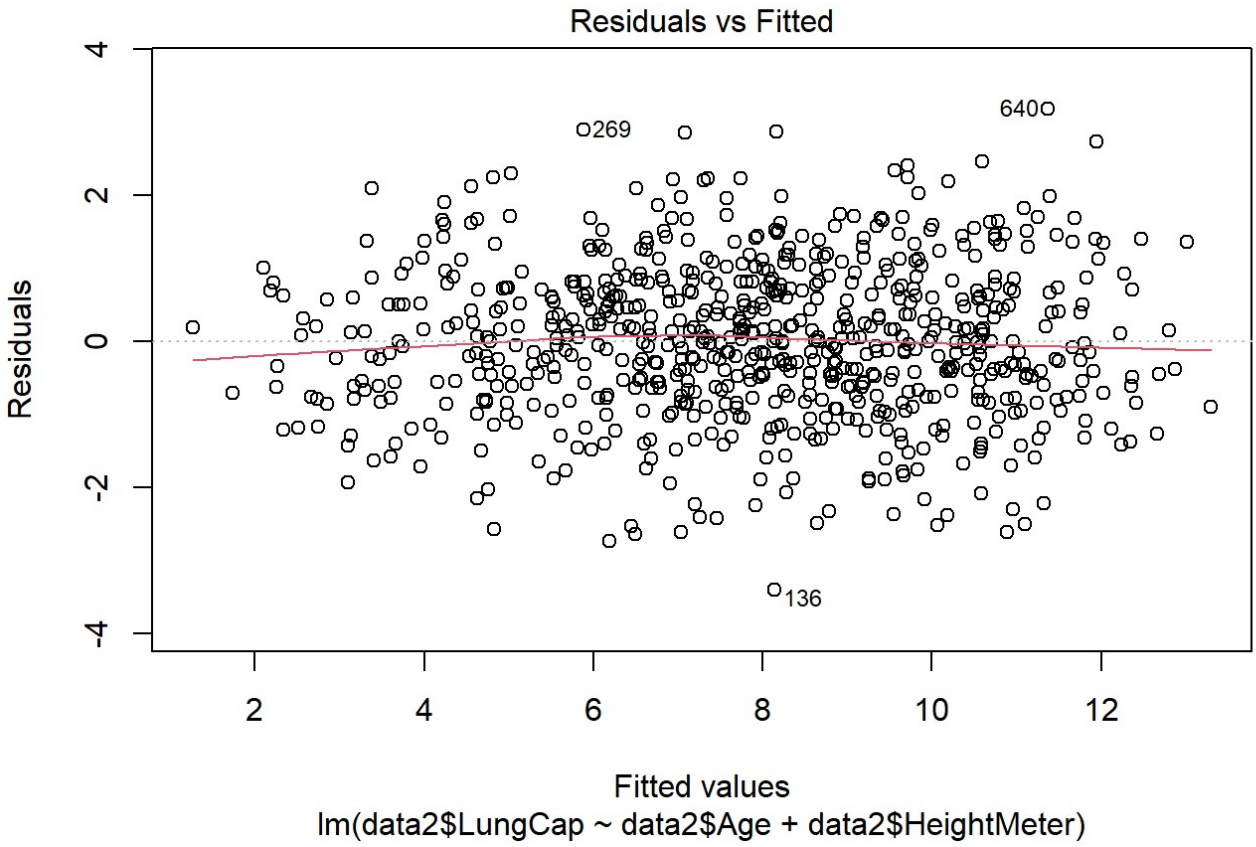
```
cor(data2$Age, data2$HeightMeter, method = "pearson")
```

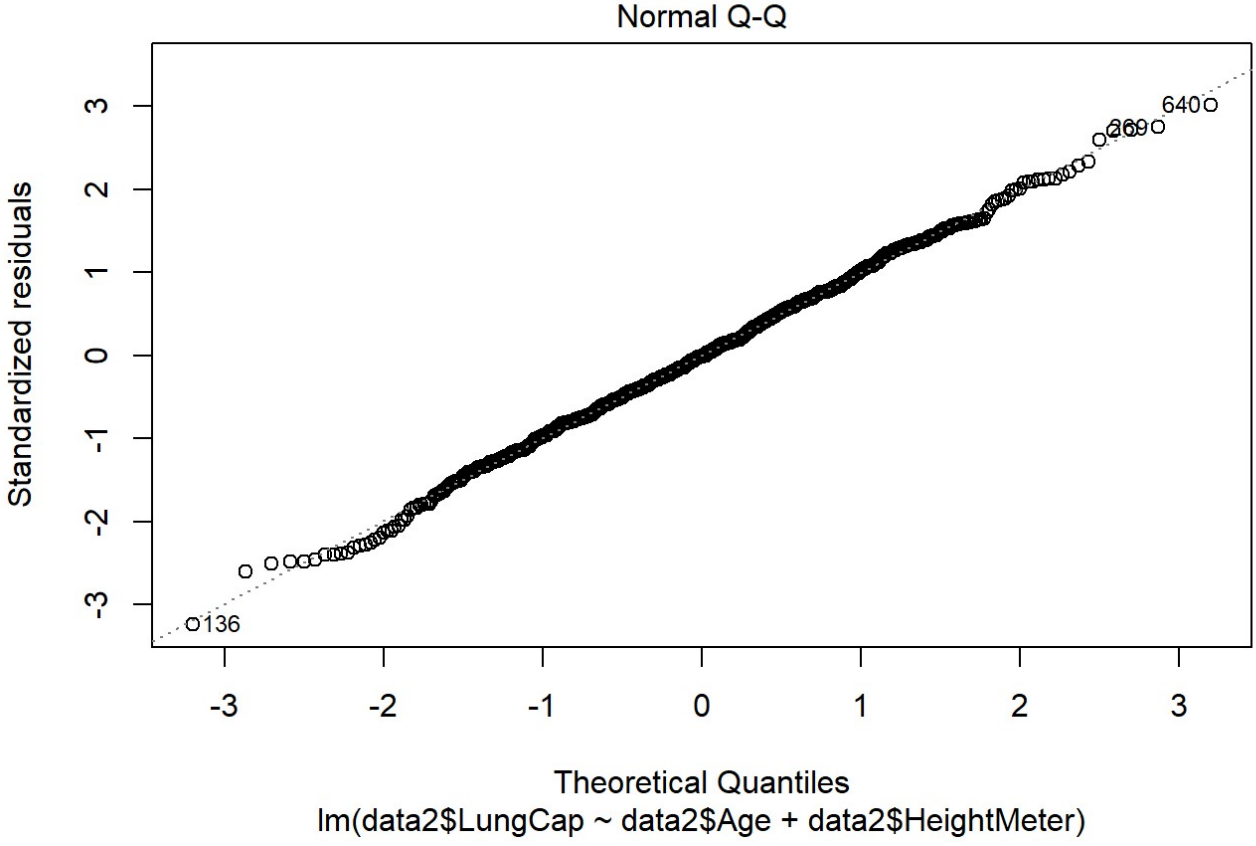
```
## [1] 0.8348995
```

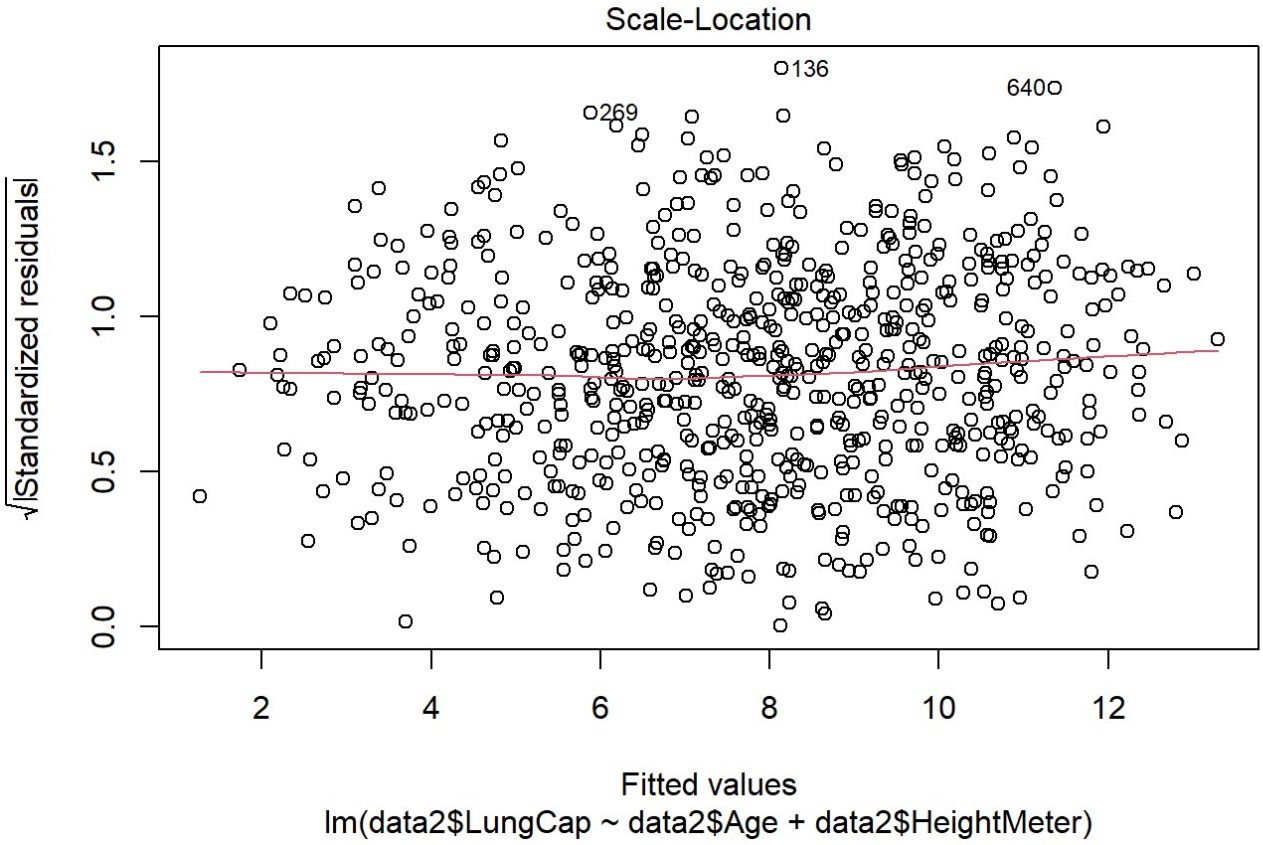
```
confint(model1, conf.level=0.95)
```

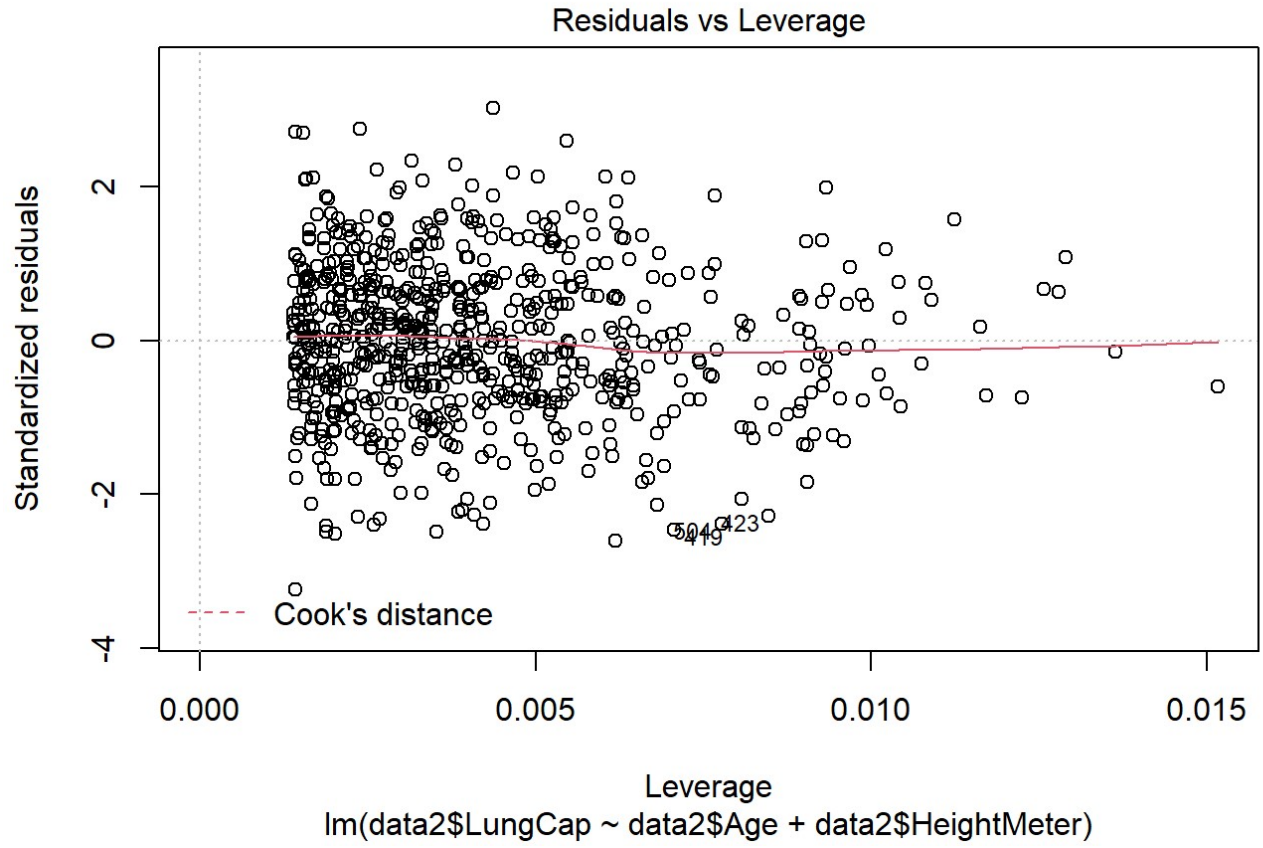
```
##                         2.5 %      97.5 %
## (Intercept)       -12.66016395 -10.7933154
## data2$Age           0.08910293   0.1590719
## data2$HeightMeter  10.20394027  11.7335008
```

```
plot(model1)
```


Residuals vs Fitted
Residuals
Fitted values
lm(data2$LungCap ~ data2$Age + data2$HeightMeter)
269
640
136


Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(data2$LungCap ~ data2$Age + data2$HeightMeter)
640
269
136





From the summary I can see that the R-squared is 0.8425 which means that approx. 84 % of variation in lung capacity can be explained by this model using age and height in meter. The residual standard error is 1.053 on 721 degrees of freedom. This gives me an idea of how far observed lung capacity (Y values) are

form the predicted or fitted lung capacity (the Y-hats). The Intercept of -11.72674 is the estimated mean Y (lung capacity) value when all X-values (age and height in meter) are zero. In other words it is the estimated mean lung capacity for a person with age and height in meter of zero. The value for age estimated is 0.12409. This is the effect of age on lung capacity adjusting or controlling for height in meter. In other words an increase of 1 year in age is associated with an increase of 0.12409 in lung capacity adjusting or controlling for height in meter. The slope for height in meter is 10.96872. This is the estimated effect of height in meter on lung capacity adjusting for age.

The Pearson's correlation between age and height in meter is 0.8348995 so the two variables are very highly correlated. The collinearity between age and height in meter means that I should not directly interpret the slope (e.g. age), as the effect of age on lung capacity adjusting for height in meter. The high correlation between age and height in meter suggests that these two effects are somewhat bounded together.

The estimated slope for age is 0.12409 or 95 % confident that the true slope is between 0.08910293 and 0.1590719. The estimated slope for height in meter is 10.96872 or 95 % confident that the true slope is between 10.20394027 and 11.7335008.

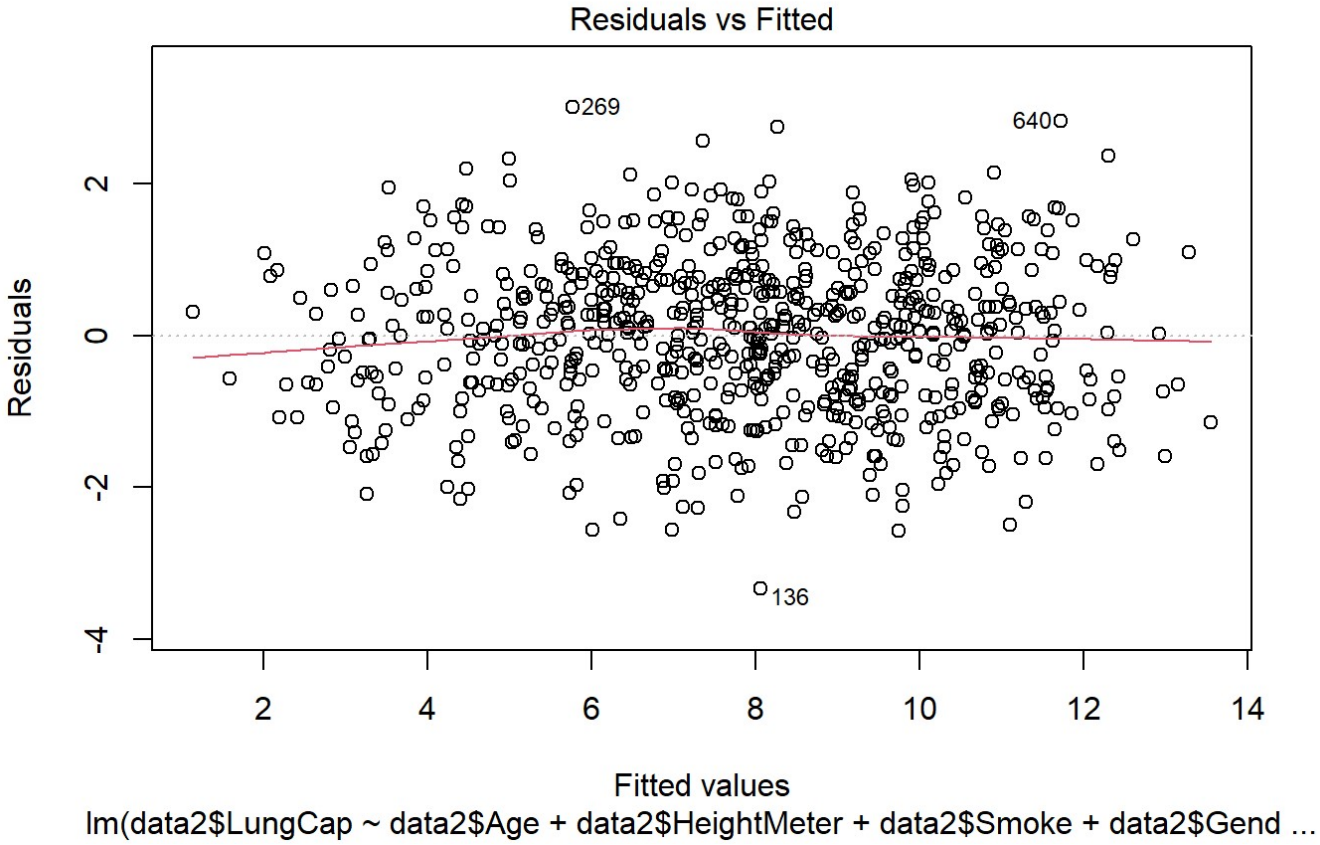
To check the regression diagnostics for this model with two independent variables I'll look at the plot for residuals vs. fitted. The relationship between lung capacity and age and height in meter is approx. linear. The variation looks constant. The lung capacity given age and height in meter is approx. normal which can been seen on the Normal Q-Q plot.

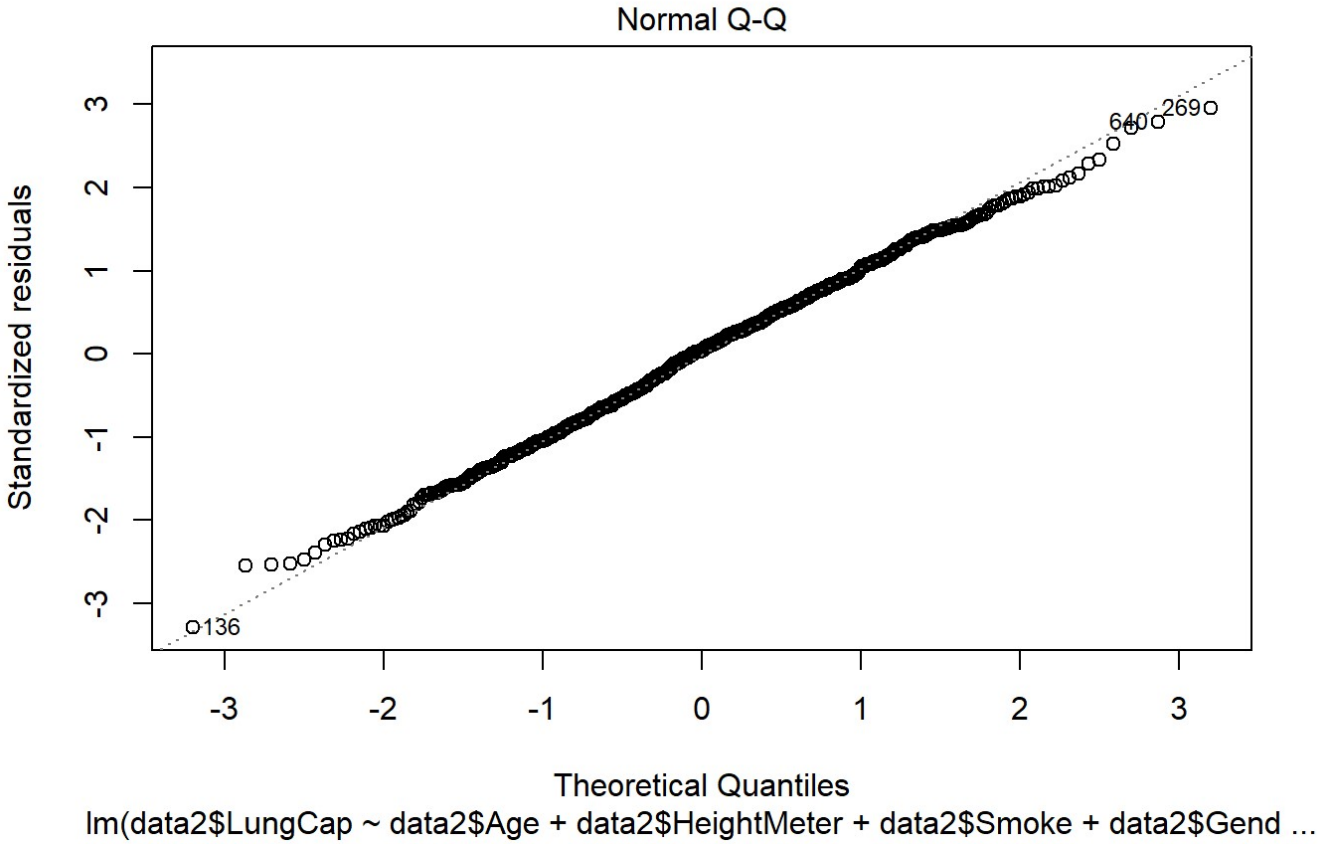
If I wanted to fit a model with all the possible X-variables that would be age, height in meter, smoke, gender and caesarean, this could be done so.

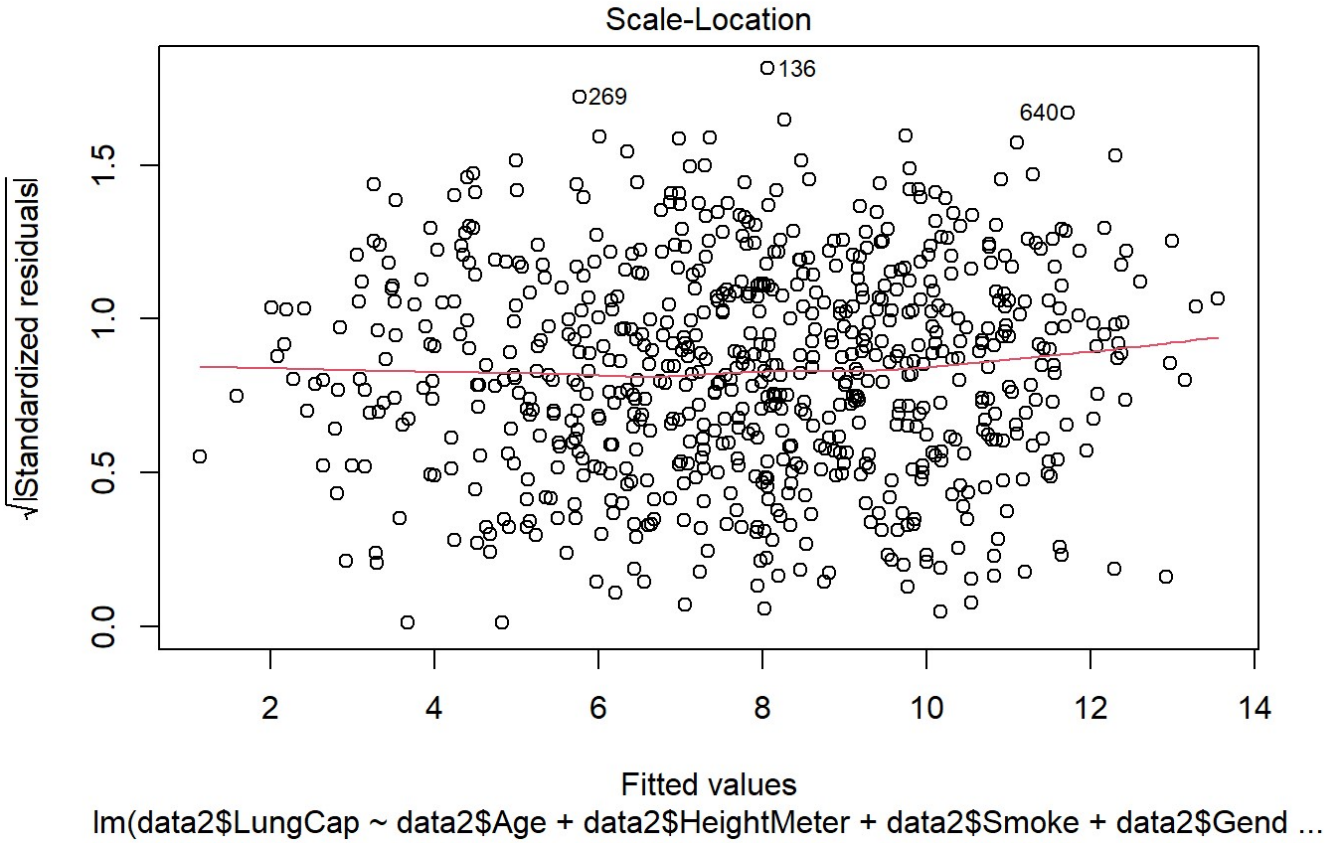
```
model2 <- lm(data2$LungCap ~ data2$Age + data2$HeightMeter + data2$Smoke + data2$G
ender + data2$Caesarean)
summary(model2)
```

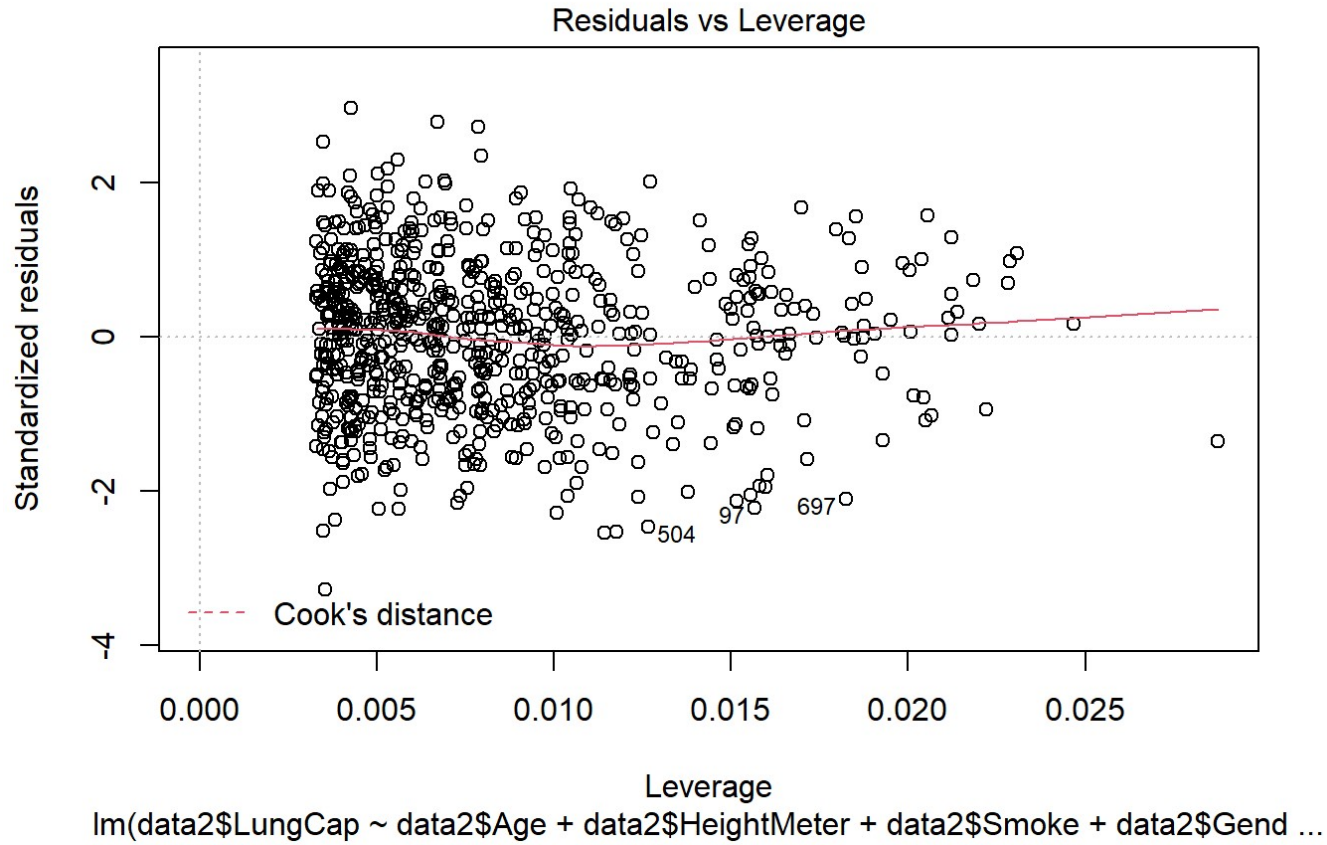
```
##
## Call:
## lm(formula = data2$LungCap ~ data2$Age + data2$HeightMeter +
##     data2$Smoke + data2$Gender + data2$Caesarean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3409 -0.7164  0.0410  0.7014  3.0080
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -11.31501    0.46995 -24.077  < 2e-16 ***
## data2$Age            0.15814    0.01801   8.783  < 2e-16 ***
## data2$HeightMeter   10.41363    0.39534  26.341  < 2e-16 ***
## data2$Smokeyes      -0.60922    0.12570  -4.847 1.54e-06 ***
## data2$Gendermale     0.37962    0.07956   4.771 2.22e-06 ***
## data2$Caesareanyes  -0.20042    0.09079  -2.208   0.0276 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.018 on 718 degrees of freedom
## Multiple R-squared:  0.8535, Adjusted R-squared:  0.8525
## F-statistic: 836.9 on 5 and 718 DF,  p-value: < 2.2e-16
```

```
plot(model2)
```


Residuals vs Fitted
269
640
136
Residuals
2
0
-2
-4
2
4
6
8
10
12
14
Fitted values
lm(data2$LungCap ~ data2$Age + data2$HeightMeter + data2$Smoke + data2$Gend ...


Normal Q-Q
640
269
136
Standardized residuals
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3
Theoretical Quantiles
lm(data2$LungCap ~ data2$Age + data2$HeightMeter + data2$Smoke + data2$Gend ...





The two models is close to each other but the multiple R-squared is higher for the model with all the X-variables. Also the residual standard error value is lower for the model with all the X-values. This makes good sense since a model provided with more data points should be able to produce a more correct

estimate.

Stage 5 - Training and testing regression model

In the following I will first split my data set into a training set and testing set. I will train my regression models on the training set and then I will test my models on the test set, and evaluate my models for how well they are to predict unknown (future) values.

Splitting the data set

In the following I will split my data set into a training- and testing set. The training set is used for training the model for predicting values and the testing set is used to test how well the model actually predicts the actual values in the testing set. It is worth to understand that the values in the testing set has not been shown to the model before.
I am splitting my data set randomly into the training set which will consist of 80 % of the original data set and the last 20 % will be the testing set. It's important to never ever train the models on the testing data set since it will skew the results and introduce bias into the results.
First I will create random numbers uniformly distributed between 0-1 for each observation in the data set. The random numbers will be used to reorder my dataset which is saved as a new dataset. To be able to reproduce this random outcome I have set the seed.

```
set.seed(123)
r <- runif(nrow(data2))
data2r <- data2[order(r), ]
train <- data2r[1:580, ]
test <- data2r[581:724, ]
str(train)
```

```
## 'data.frame':    580 obs. of  7 variables:
##  $ LungCap    : num  8.43 7.83 7.17 8.7 8.82 ...
##  $ Age        : int  16 13 17 14 16 8 10 18 9 16 ...
##  $ Height     : num  70.5 71 68.8 73.1 71.3 58.4 59.7 68.6 62.4 74.3 ...
##  $ Smoke      : chr  "yes" "no" "no" "no" ...
##  $ Gender     : chr  "male" "female" "male" "male" ...
##  $ Caesarean  : chr  "no" "no" "yes" "yes" ...
##  $ HeightMeter: num  1.79 1.8 1.75 1.86 1.81 ...
```

```
str(test)
```

```
## 'data.frame':    144 obs. of  7 variables:
##  $ LungCap    : num  3.9 9.68 8.43 9.2 5.38 ...
##  $ Age        : int  10 12 12 14 11 10 10 5 11 12 ...
##  $ Height     : num  58 70.9 69.2 68.7 59.3 65.4 63.2 48.9 63 60.3 ...
##  $ Smoke      : chr  "yes" "no" "no" "no" ...
##  $ Gender     : chr  "female" "male" "male" "female" ...
##  $ Caesarean  : chr  "no" "no" "no" "yes" ...
##  $ HeightMeter: num  1.47 1.8 1.76 1.74 1.51 ...
```

The training set contains 580 observations and the test set contains 144 observations.

Training the model

In the following I will train different regression models on the training set. I will at first train a model with one parameter and thereafter two models with multiple parameters.

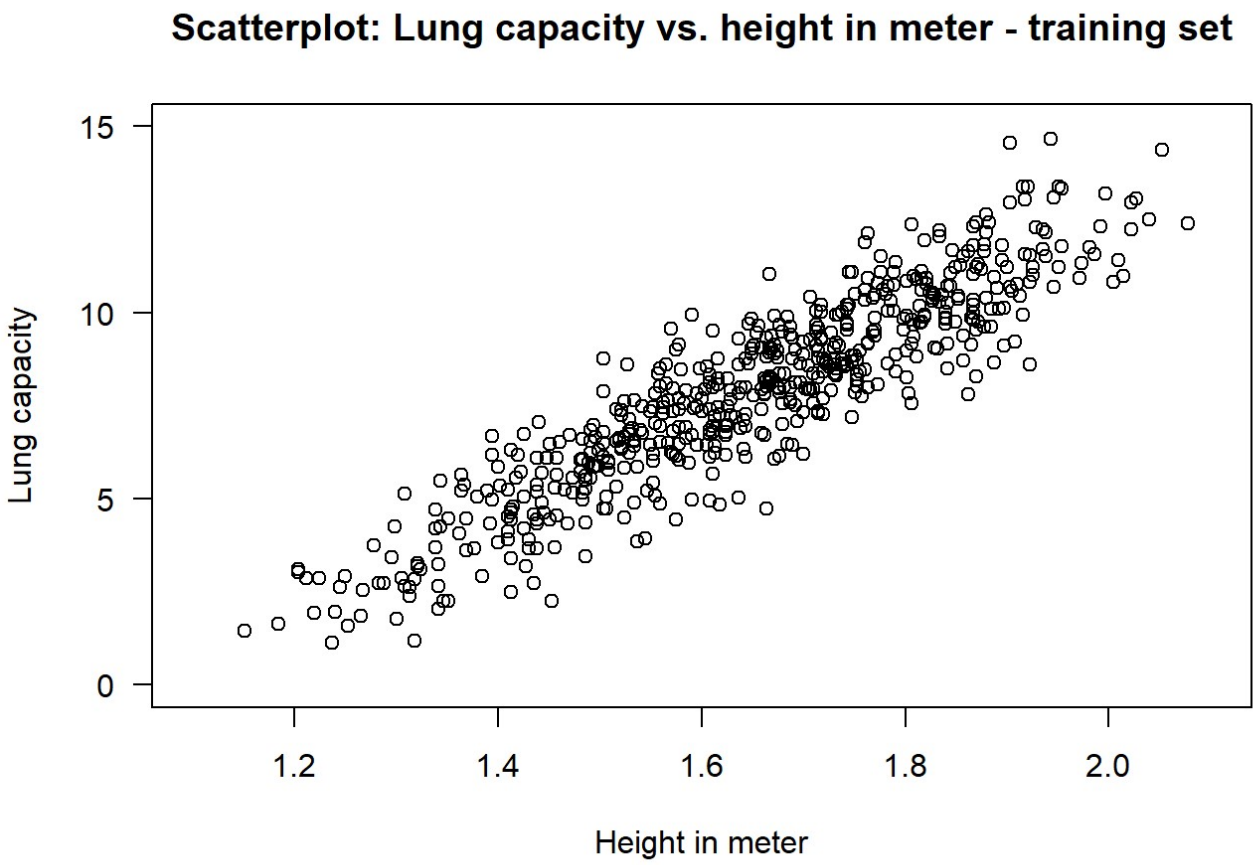
```
cor(train$LungCap,train$HeightMeter)
```

```
## [1] 0.9099666
```

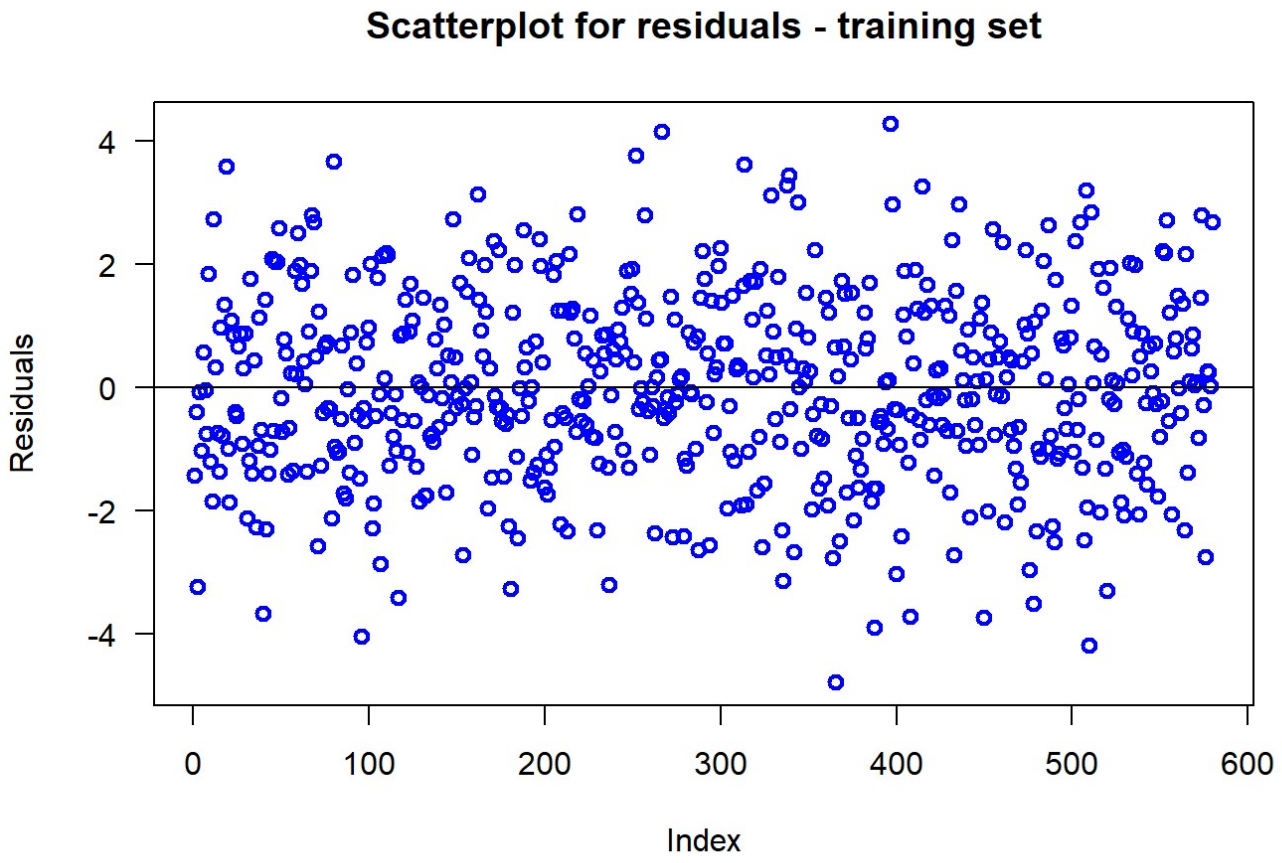
```
fit_model1 <- lm(LungCap~Age, data = train)
summary(fit_model1)
```

```
##
## Call:
## lm(formula = LungCap ~ Age, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7840 -0.9950 -0.0175  0.9738  4.2590
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.17515    0.20435   5.751 1.44e-08 ***
## Age          0.54299    0.01576  34.458  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.515 on 578 degrees of freedom
## Multiple R-squared:  0.6726, Adjusted R-squared:  0.672
## F-statistic:  1187 on 1 and 578 DF,  p-value: < 2.2e-16
```

```
plot(train$HeightMeter, train$LungCap, main ="Scatterplot: Lung capacity vs. heigh
t in meter - training set", ylab = "Lung capacity", xlab = "Height in meter", las=
1, xlim=c(1.1,2.1), ylim=c(0,15))
```




```
plot(fit_model1$residuals, col="blue", lwd=2, ylab="Residuals", las=1, main = "Sca
tterplot for residuals - training set")
abline(h=0)
```



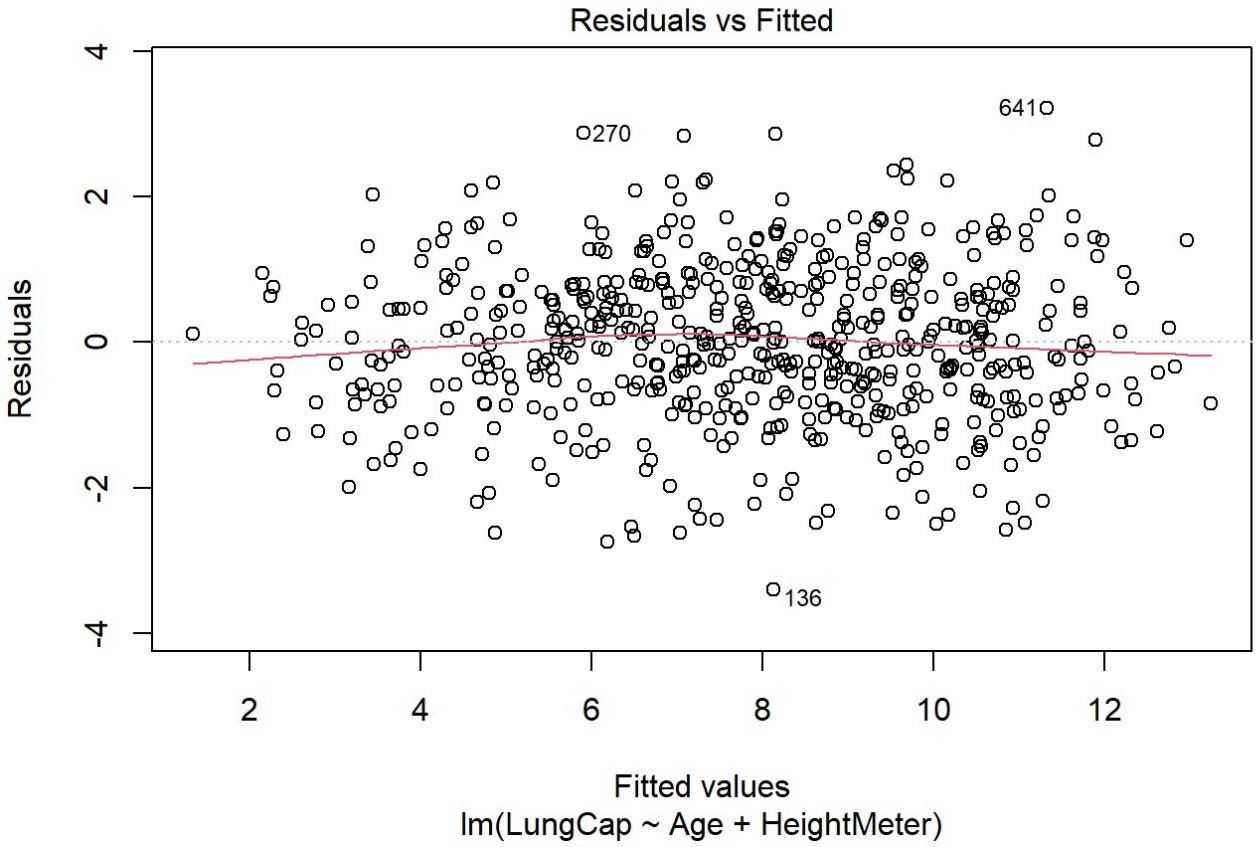
From now on I will refer to the above model as model 1. When I look at the correlation coefficient between lung capacity and height in meter there seems to be a very strong correlation. The scatterplot indicates a strong, positive, linear relationship between the lung capacity and the height in meter. The scatterplot for the residuals shows how the residuals are randomly distributed. The residual standard error is 1.515 and the R-squared is 0.6726 which is approx. 67 % which in a moment will be compared to the other linear regression models.

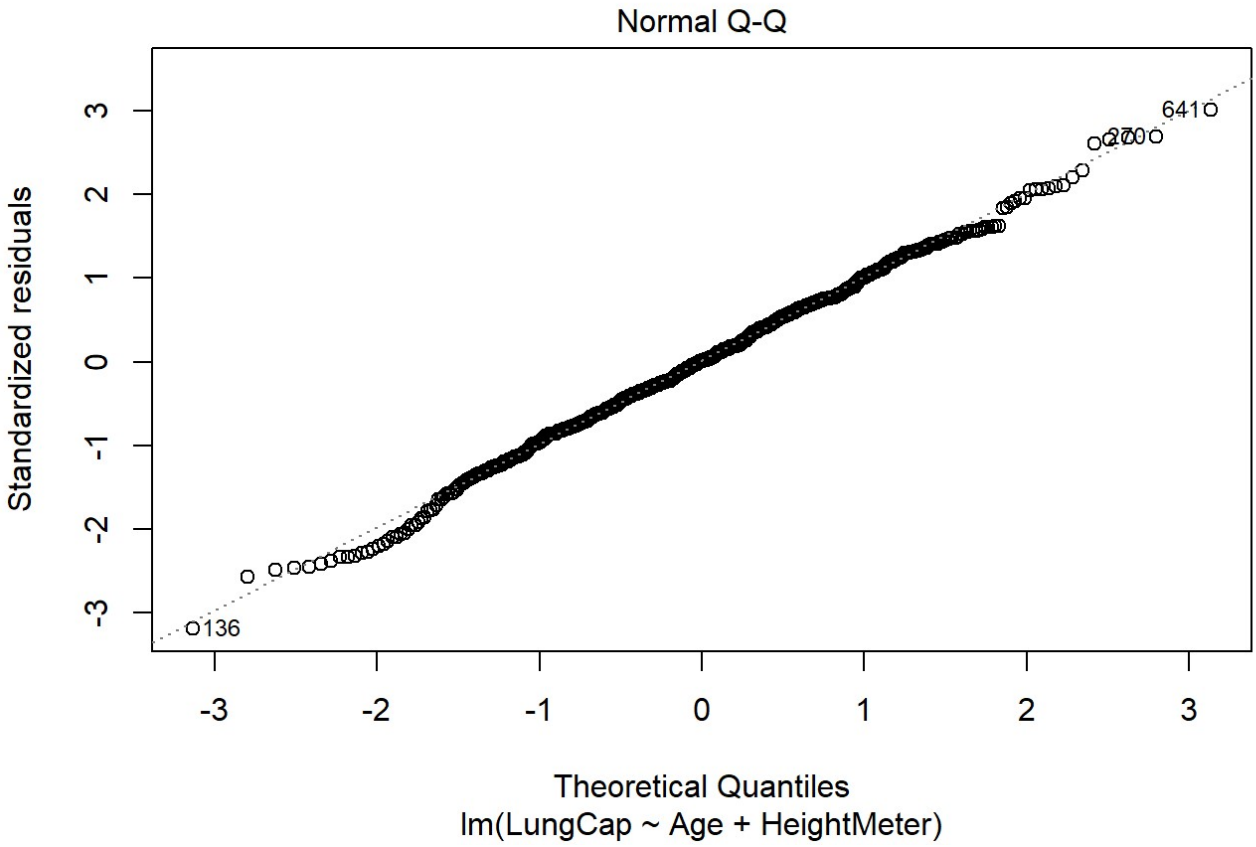
Now I will train a regression model with multiple variables as both height in meter and age as the x-variables. I will refer to this model as model 2.

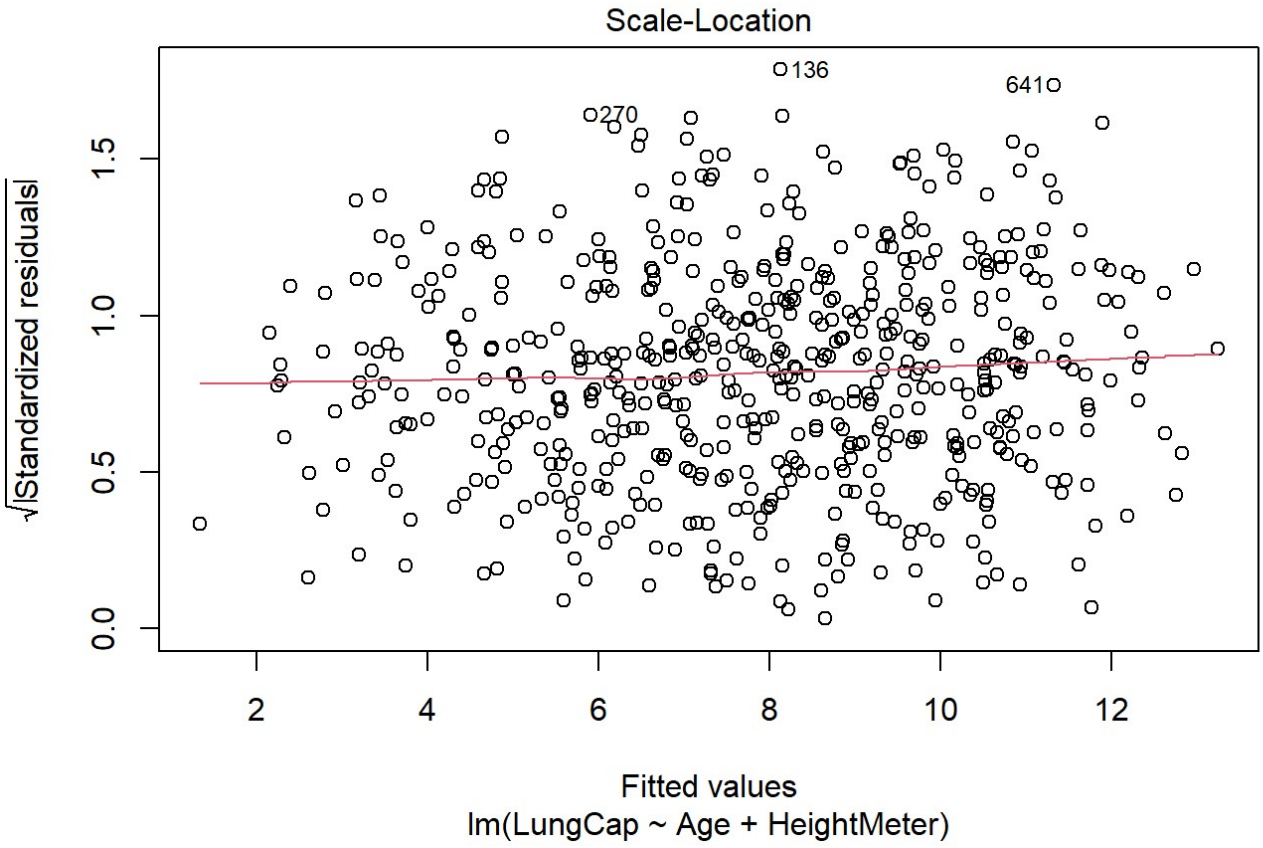
```
fit_model2 <- lm(LungCap~Age+HeightMeter, data = train)
summary(fit_model2)
```

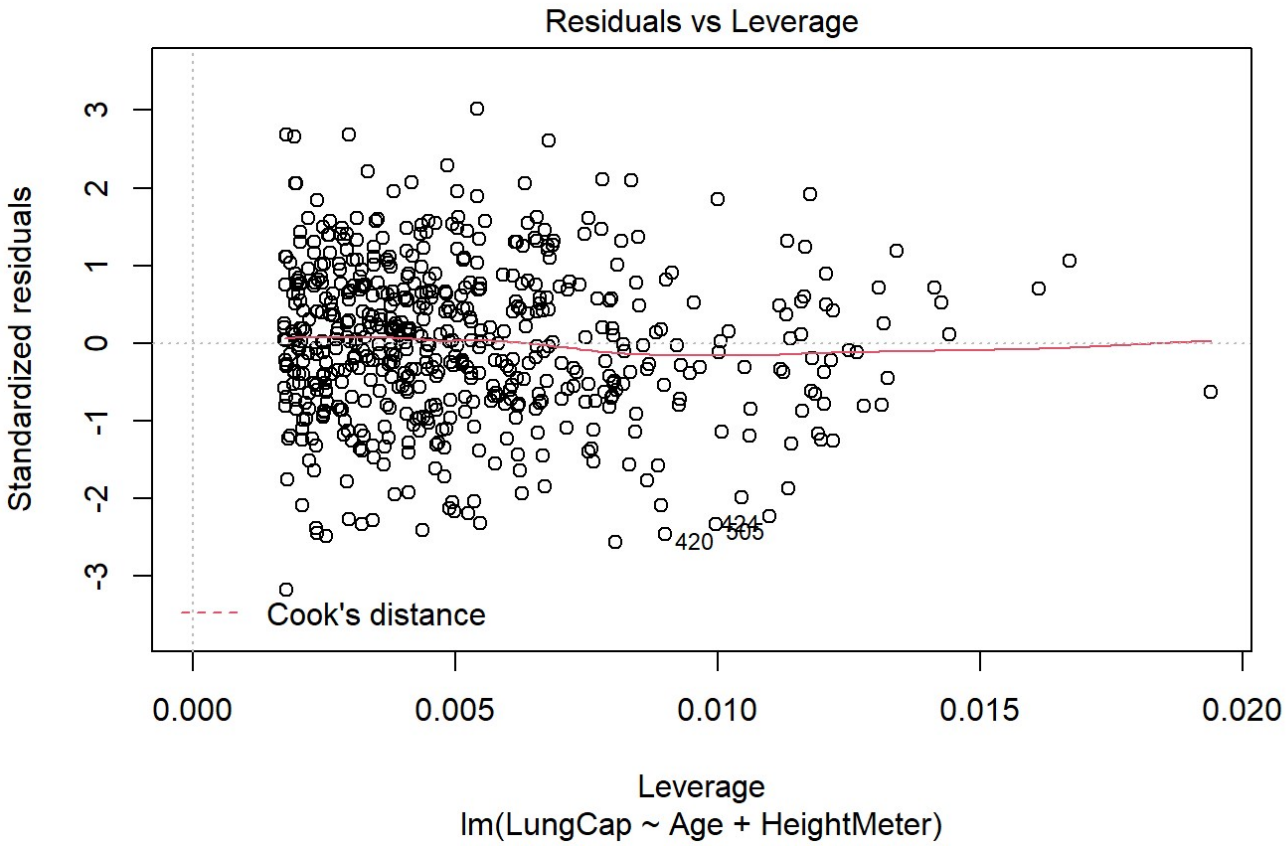
```
##
## Call:
## lm(formula = LungCap ~ Age + HeightMeter, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4047 -0.6953  0.0086  0.7365  3.2158
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -11.59625    0.54762 -21.176  < 2e-16 ***
## Age           0.11919    0.02076   5.742 1.51e-08 ***
## HeightMeter  10.92532    0.45194  24.174  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.069 on 577 degrees of freedom
## Multiple R-squared:  0.8373, Adjusted R-squared:  0.8368
## F-statistic:  1485 on 2 and 577 DF,  p-value: < 2.2e-16
```

```
plot(fit_model2)
```




Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(LungCap ~ Age + HeightMeter)
136
270
641


Scale-Location
√|Standardized residuals|
Fitted values
lm(LungCap ~ Age + HeightMeter)
136
270
641

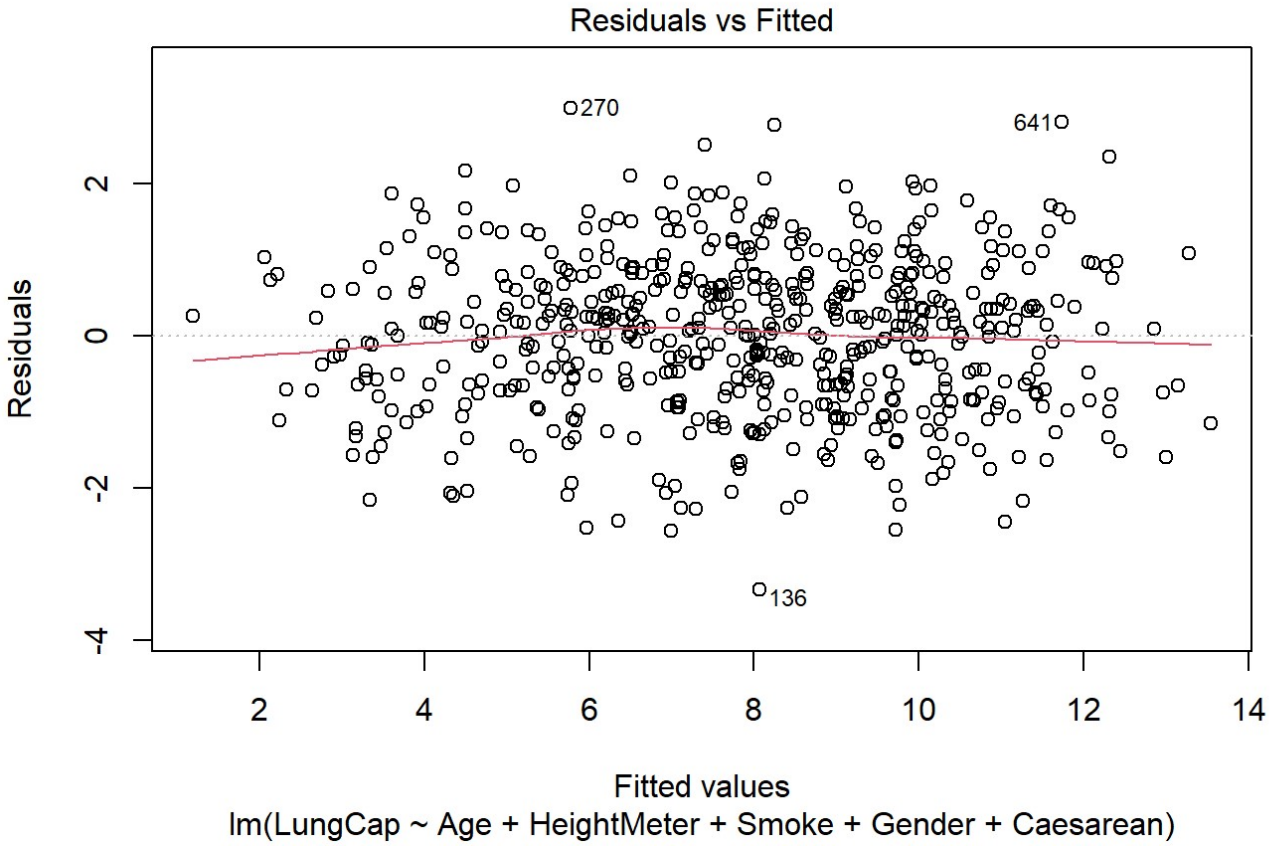


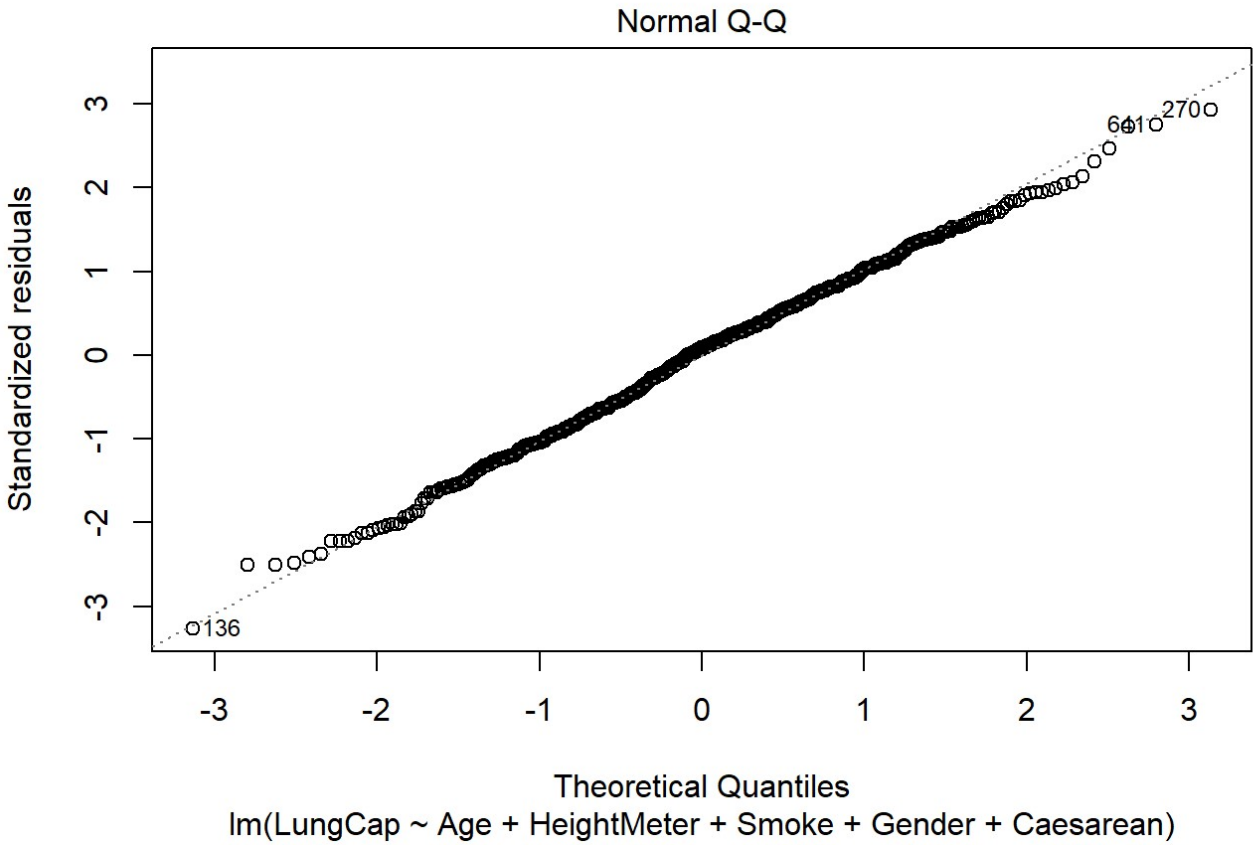
From model 2 the main point for now is to look at the residual standard error, which is 1.069 and the R-squared is 0.8373 which is approx. 84 %. As can be seen on the residuals vs. fitted scatterplot the residuals are randomly distributed. The residual standard error and the R-squared will be discussed in a moment after having trained the last linear model with multiple variables which will be age, height in meter, smoke, gender and caesarean. I will refer to this model as model 3.

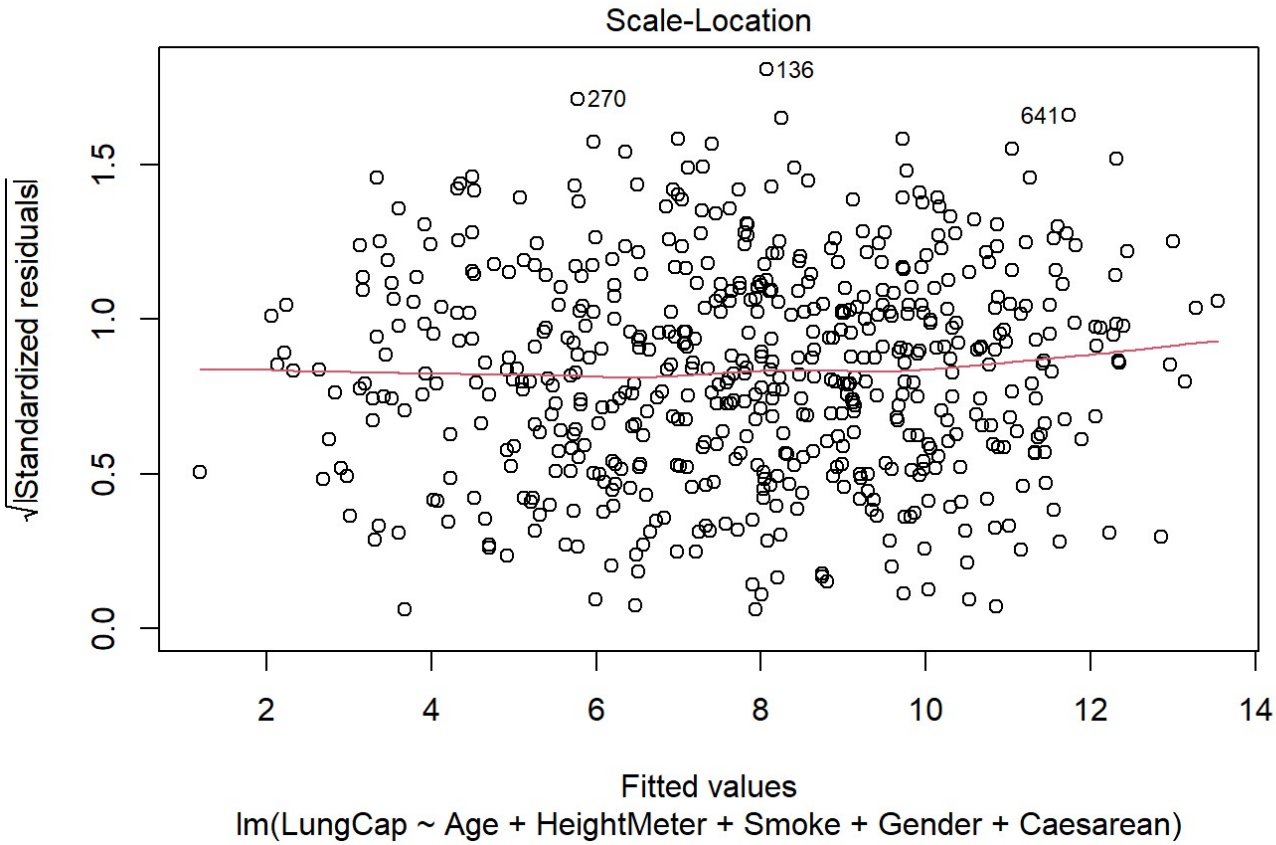
```
fit_model3 <- lm(LungCap~Age+HeightMeter+Smoke+Gender+Caesarean, data = train)
summary(fit_model3)
```

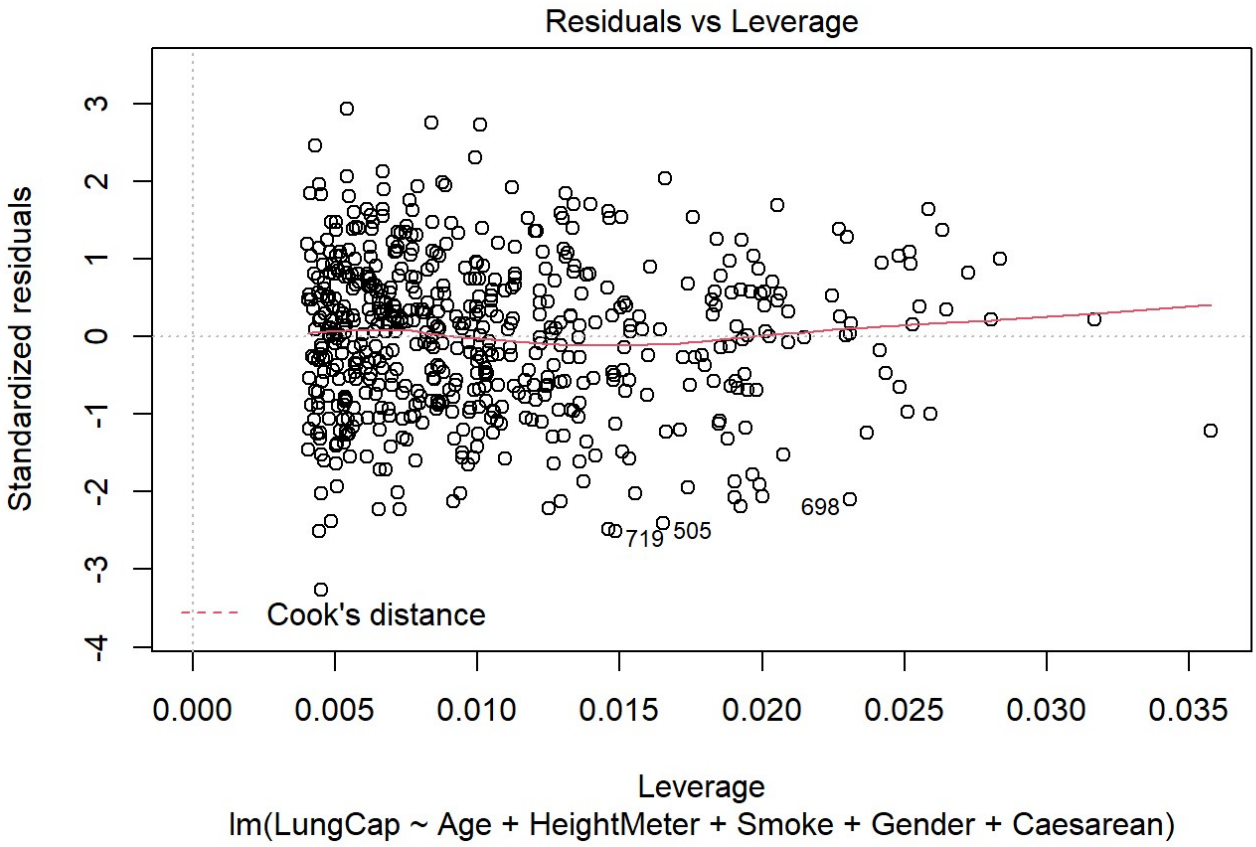
```
##
## Call:
## lm(formula = LungCap ~ Age + HeightMeter + Smoke + Gender + Caesarean,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3387 -0.7135  0.0804  0.6984  2.9957
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -11.12298    0.53836 -20.661  < 2e-16 ***
## Age            0.15967    0.02087   7.651 8.47e-14 ***
## HeightMeter   10.28495    0.45625  22.543  < 2e-16 ***
## Smokeyes      -0.65619    0.14085  -4.659 3.96e-06 ***
## Gendermale     0.42259    0.08985   4.703 3.21e-06 ***
## Caesareanyes  -0.26673    0.10214  -2.611  0.00926 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.024 on 574 degrees of freedom
## Multiple R-squared:  0.8515, Adjusted R-squared:  0.8502
## F-statistic: 658.4 on 5 and 574 DF,  p-value: < 2.2e-16
```

```
plot(fit_model3)
```




Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(LungCap ~ Age + HeightMeter + Smoke + Gender + Caesarean)
136
641
270


Scale-Location
√|Standardized residuals|
Fitted values
lm(LungCap ~ Age + HeightMeter + Smoke + Gender + Caesarean)
136
270
641



For model 3 the residual standard error is 1.024 and the R-squared is 0.8515 which is approx. 85 %. Compared with the other two models based on the training set, model 3 has the lowest error and highest percentage of R-squared which means that it explains 85 % of the dependent variable variation. Then follows model 2 and at last comes model 1.

Testing the model

In the following I will test my models and evaluate how well they perform. To assess my models performance on unseen data I have trained them on the training data set and then I use the testing data set to compare predicted values with actual values. If the model is not tested and is made such that it performs good only on the training data set, then the parameters will be tuned in a way that they are only good enough to predict the value for the data which was in the training data set, and that is not generically applicable to any other data sets that I want to carry the model over to and that is called overfitting. I want to avoid overfitting and underfitting my models.

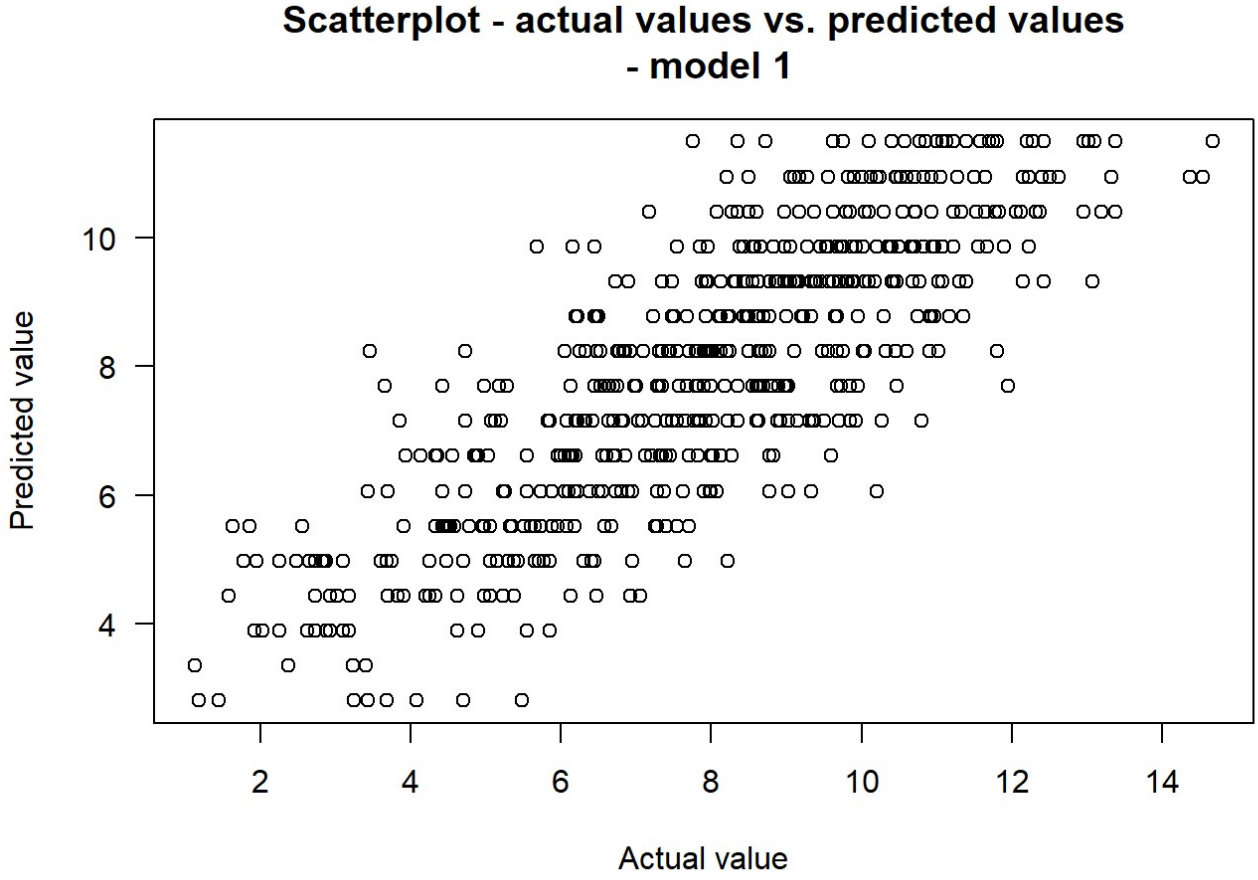
RMSE

I will use RMSE (root mean squared error) to evaluate my models. When I use my models I will be able to compare the predicted values to actual values. In the following I am creating a new dataset based on the training set and I have added a column for each of the models with the predicted value for lung capacity.

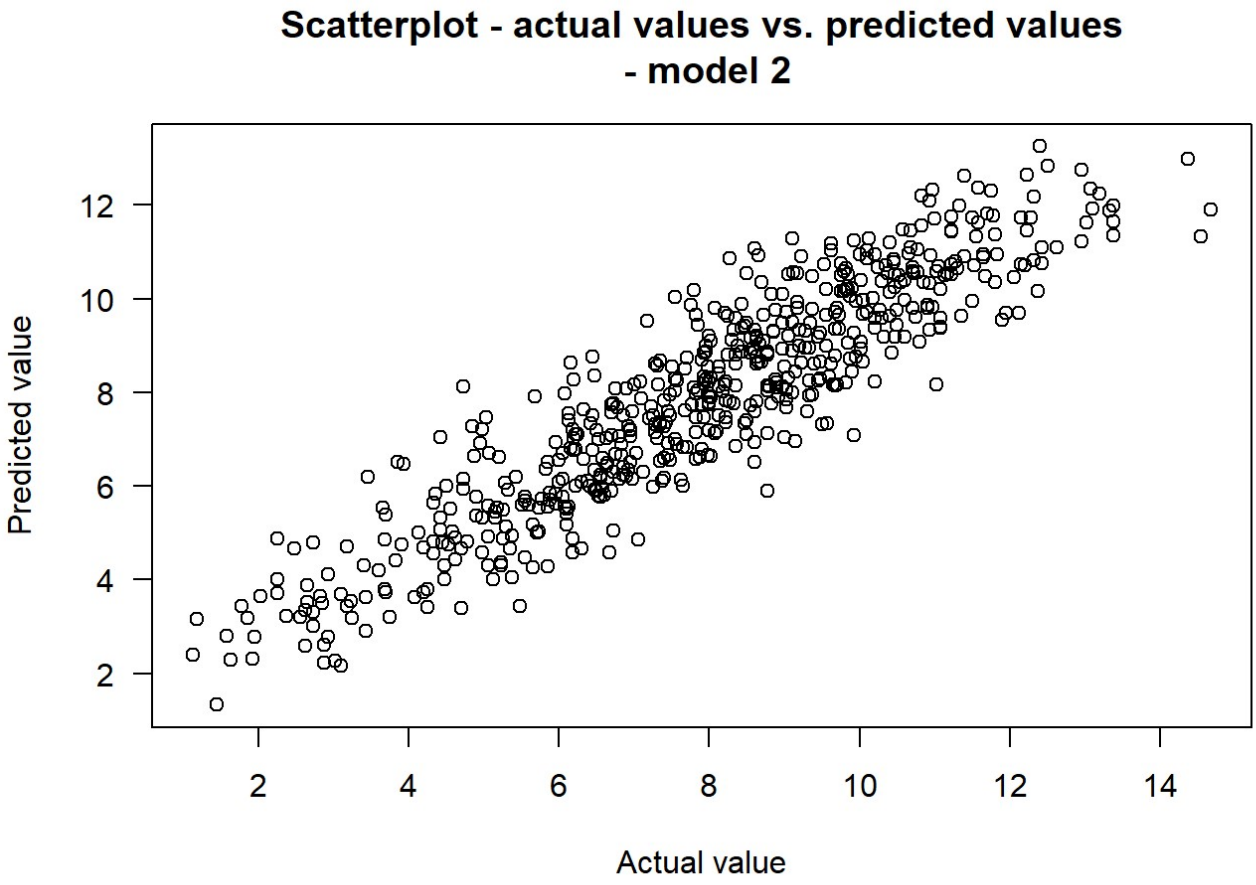
```
# model 1
lungCap_mod_rmse_train <- train %>%
   mutate(predModel1Train = predict(fit_model1))
# model 2
lungCap_mod_rmse_train$predModel2Train = predict(fit_model2)
# model 3
lungCap_mod_rmse_train$predModel3Train = predict(fit_model3)
```

I will look at the actual values plotted against the predicted values.

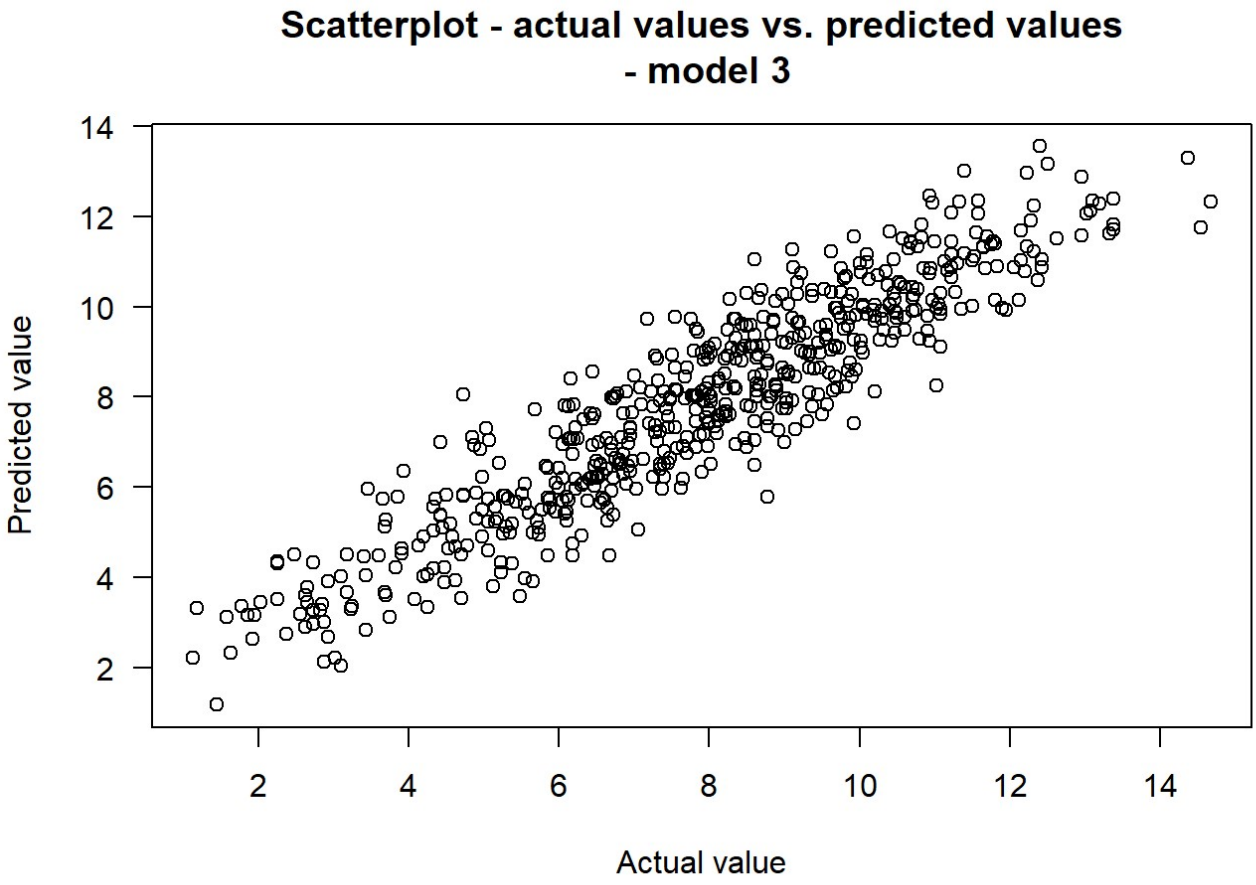
```
plot(lungCap_mod_rmse_train$LungCap, lungCap_mod_rmse_train$predModel1Train, las=
1, main="Scatterplot - actual values vs. predicted values\n - model 1", ylab="Pred
icted value", xlab="Actual value")
```




```
plot(lungCap_mod_rmse_train$LungCap, lungCap_mod_rmse_train$predModel2Train, las=
1, main="Scatterplot - actual values vs. predicted values\n - model 2", ylab="Pred
icted value", xlab="Actual value")
```




```
plot(lungCap_mod_rmse_train$LungCap, lungCap_mod_rmse_train$predModel3Train, las=
1, main="Scatterplot - actual values vs. predicted values\n - model 3", ylab="Pred
icted value", xlab="Actual value")
```



All of the models shows a positive linear relationship between the actual values and predicted values. Some of the data points in model 1 could be outliers, but model 2 and 3 does not seem to have any outliers. The correlation shows that when an actual value increase so does the predicted one. When comparing the three scatterplots it is possible to observe that models with more x-variables is more dense than models with a lesser number of x-variables.

Now I will create the mean squared error for each model.

```
# model 1
mse_model1 <- lungCap_mod_rmse_train %>% mutate(errorModel1 = predModel1Train - Lu
ngCap, sqrErrorModel1 = errorModel1^2) %>% summarise(mse_model1 = mean(sqrErrorMod
el1))
# model 2
mse_model2 <- lungCap_mod_rmse_train %>% mutate(errorModel2 = predModel2Train - Lu
ngCap, sqrErrorModel2 = errorModel2^2) %>% summarise(mse_model2 = mean(sqrErrorMod
el2))
# model 3
mse_model3 <- lungCap_mod_rmse_train %>% mutate(errorModel3 = predModel3Train - Lu
ngCap, sqrErrorModel3 = errorModel3^2) %>% summarise(mse_model3 = mean(sqrErrorMod
el3))

print(paste0("Mean squared error for model 1: ",mse_model1))
```

```
## [1] "Mean squared error for model 1: 2.28795709003217"
```

```
print(paste0("Mean squared error for model 2: ",mse_model2))
```

```
## [1] "Mean squared error for model 2: 1.13669490226121"
```

```
print(paste0("Mean squared error for model 3: ",mse_model3))
```

```
## [1] "Mean squared error for model 3: 1.03749493748472"
```

In the above code chunk I first calculate the error which is the predicted value minus the actual value for each model. The sum of the errors for each model sums to zero since some of the values are positive and some of the values are negative and the average mean would be zero. By squaring the errors all the values becomes positive and then I find the average of that. When summarizing I take the mean of the squared error. To get back into my the original scale I need to take the square root of the mean squared error, which I do in the following.

```
# Model 1
RMSE_model1 <- sqrt(mse_model1)
print(paste0("RMSE for model 1 (training): ", RMSE_model1))
```

```
## [1] "RMSE for model 1 (training): 1.51259944798092"
```

```
# Model 2
RMSE_model2 <- sqrt(mse_model2)
print(paste0("RMSE for model 2 (training): ", RMSE_model2))
```

```
## [1] "RMSE for model 2 (training): 1.06615894793469"
```

```
# Model 3
RMSE_model3 <- sqrt(mse_model3)
print(paste0("RMSE for model 3 (training): ", RMSE_model3))
```

```
## [1] "RMSE for model 3 (training): 1.01857495427912"
```

All RMSE values shows a pretty fine average error and they are between 7-11 %. The RMSE values was for the training set, so I will also have to look into the testing set. Remember that the models have not seen the data in the test set and if the model is good I should get a RMSE value for each model that is close to the RMSE value from the training set.

```
#model 1
pred_reg_test_model1 <- predict(fit_model1, newdata = test)
reg_rmse_test_model1 <- sqrt(mean((pred_reg_test_model1 - test$LungCap)^2))
#model 2
pred_reg_test_model2 <- predict(fit_model2, newdata = test)
reg_rmse_test_model2 <- sqrt(mean((pred_reg_test_model2 - test$LungCap)^2))
#model 1
pred_reg_test_model3 <- predict(fit_model3, newdata = test)
reg_rmse_test_model3 <- sqrt(mean((pred_reg_test_model3 - test$LungCap)^2))

print(paste0("RMSE for model 1 (test): ", reg_rmse_test_model1))
```

```
## [1] "RMSE for model 1 (test): 1.56179818624582"
```

```
print(paste0("RMSE for model 2 (test): ", reg_rmse_test_model2))
```

```
## [1] "RMSE for model 2 (test): 0.98786586183215"
```

```
print(paste0("RMSE for model 3 (test): ", reg_rmse_test_model3))
```

```
## [1] "RMSE for model 3 (test): 0.996581262276729"
```

To get the error I took the predicted value and subtracted the actual value. Then I squared the errors to make them positive. Then I took the mean of the squared error and then I took the square root of that. So it is basically the same I did for the training set above.

Now it is time to compare the RMSE value for each model on the test and training set.

```
diff_model1 <- RMSE_model1 - reg_rmse_test_model1
diff_model2 <- RMSE_model2 - reg_rmse_test_model2
diff_model3 <- RMSE_model3 - reg_rmse_test_model3
print(paste0("The difference between training and test RMSE value on model 1: ",ab
s(diff_model1)))
```

```
## [1] "The difference between training and test RMSE value on model 1: 0.04919873
82649006"
```

```
print(paste0("The difference between training and test RMSE value on model 2: ",ab
s(diff_model2)))
```

```
## [1] "The difference between training and test RMSE value on model 2: 0.07829308
61025407"
```

```
print(paste0("The difference between training and test RMSE value on model 3: ",ab
s(diff_model3)))
```

```
## [1] "The difference between training and test RMSE value on model 3: 0.02199369
20023942"
```

When looking at the difference in RMSE value all three models have a very small difference value. That means that the models performs well on both the training and the test set. So the models could be used on other unseen data sets to predict the lung capacity. Model 2 has the largest difference between the RMSE values for test and training set. After that follows model 1. Model 3 has the lowest difference and would therefor be the best model to use on unseen dataset. Model 2 has a larger difference than model 1 which could be explained by an overfitting of the model. However it has to be noted that even model 2 has a very small difference between the training and test set.
The RMSE value is good if needed to adjust for large, rare errors.

MAE

In the following I will compute the mean absolute error for all three models, both on the training and the test set. Then I will look at the difference between the MAE value for the training and the test set on each model.

```
# model 1
mae_train_model1 <- MAE(lungCap_mod_rmse_train$predModel1Train, lungCap_mod_rmse_t
rain$LungCap)
mae_test_model1 <- MAE(pred_reg_test_model1, test$LungCap)
print(paste0("MAE on training set model 1: ",mae_train_model1))
```

```
## [1] "MAE on training set model 1: 1.20605040572097"
```

```
print(paste0("MAE on testing set model 1: ",mae_test_model1))
```

```
## [1] "MAE on testing set model 1: 1.26311868593629"
```

```
print(paste0("The difference between training and test MAE value on model 1: ",abs
(mae_train_model1-mae_test_model1)))
```

```
## [1] "The difference between training and test MAE value on model 1: 0.057068280
2153235"
```

```
# model 2
mae_train_model2 <- MAE(lungCap_mod_rmse_train$predModel2Train, lungCap_mod_rmse_t
rain$LungCap)
mae_test_model2 <- MAE(pred_reg_test_model2, test$LungCap)
print(paste0("MAE on training set model 2: ",mae_train_model2))
```

```
## [1] "MAE on training set model 2: 0.847955848169728"
```

```
print(paste0("MAE on testing set model 2: ",mae_test_model2))
```

```
## [1] "MAE on testing set model 2: 0.797822584768878"
```

```
print(paste0("The difference between training and test MAE value on model 2: ",abs
(mae_train_model2-mae_test_model2)))
```

```
## [1] "The difference between training and test MAE value on model 2: 0.050133263
4008498"
```

```
# model 3
mae_train_model3 <- MAE(lungCap_mod_rmse_train$predModel2Train, lungCap_mod_rmse_t
rain$LungCap)
mae_test_model3 <- MAE(pred_reg_test_model3, test$LungCap)
print(paste0("MAE on training set model 3: ",mae_train_model3))
```

```
## [1] "MAE on training set model 3: 0.847955848169728"
```

```
print(paste0("MAE on testing set model 3: ",mae_test_model3))
```

```
## [1] "MAE on testing set model 3: 0.821501429212382"
```

```
print(paste0("The difference between training and test MAE value on model 3: ",abs
(mae_train_model3-mae_test_model3)))
```

```
## [1] "The difference between training and test MAE value on model 3: 0.026454418
9573456"
```

The absolute error is the absolute value of the difference between the forecasted value and the actual value. MAE tells how big of an error to expect from the forecast on average.[3] The difference between the MAE value on training and test set is fairly small. Model 3 has the smallest difference, followed by model 2 and then model 1. Again model 3 would be the preferred model to use on unseen data sets, but the other models would be acceptable as well.
Since model 1 has a MEA value on the test set of 1.26 it will overestimate the value when it is forecasting a value. Model 2 has a MAE value of 0.80 and model 3 has a MAE value of 0.81. Model 3 would be the prefered model to use, but the difference between model 3 and model 2 is fairly small.

MAPE

One problem with the MAE is that the relative size of the error is not always obvious. Sometimes it is hard to tell a big error from a small error. To deal with this problem, I can find the mean absolute error in percentage terms. Mean Absolute Percentage Error (MAPE) allows me to compare forecasts of different series in different scales.[4]
In the following I will compute the MAPE value for each model. This is done by finding the error by subtracting the predicted value from the actual value. Then I find the absolute value of the error and divide it by the actual value and multiply it with 100. I sum up all the absolute errors in percent and divide them by the number of rows, which will give me the average in percentage.
I do this for each model on both the training and the testing set.

```
# model 1 train
actual <- train$LungCap
predicted <- lungCap_mod_rmse_train$predModel1Train
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 580
mape_model1_train <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 1 on the training set is: ", mape_m
odel1_train))
```

```
## [1] "The MAPE in percentage for model 1 on the training set is: 19.911105336965
6"
```

```
# model 1 test
actual <- test$LungCap
predicted <- pred_reg_test_model1
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 144
mape_model1_test <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 1 on the test set is: ", mape_model
1_test))
```

```
## [1] "The MAPE in percentage for model 1 on the test set is: 21.173937217079"
```

```
print(paste0("The difference between the MAPE value on test and training set for m
odel 1 is ", abs(mape_model1_train - mape_model1_test)))
```

```
## [1] "The difference between the MAPE value on test and training set for model 1
is 1.26283188011337"
```

```
# model 2 train
actual <- train$LungCap
predicted <- lungCap_mod_rmse_train$predModel2Train
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 580
mape_model2_train <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 2 on the training set is: ", mape_m
odel2_train))
```

```
## [1] "The MAPE in percentage for model 2 on the training set is: 13.240764792810
5"
```

```
# model 2 test
actual <- test$LungCap
predicted <- pred_reg_test_model2
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 144
mape_model2_test <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 2 on the test set is: ", mape_model
2_test))
```

```
## [1] "The MAPE in percentage for model 2 on the test set is: 12.7824721527121"
```

```
print(paste0("The difference between the MAPE value on test and training set for m
odel 2 is ", abs(mape_model2_train - mape_model2_test)))
```

```
## [1] "The difference between the MAPE value on test and training set for model 2
is 0.458292640098373"
```

```
# model 3 train
actual <- train$LungCap
predicted <- lungCap_mod_rmse_train$predModel3Train
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 580
mape_model3_train <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 3 on the training set is: ", mape_m
odel3_train))
```

```
## [1] "The MAPE in percentage for model 3 on the training set is: 12.988983527750
3"
```

```
# model 3 test
actual <- test$LungCap
predicted <- pred_reg_test_model3
error <- actual - predicted
absolute_error <- abs(error)
absolute_persent_error <- absolute_error / actual * 100
sum_of_column <- sum(absolute_persent_error)
num_of_rows <- 144
mape_model3_test <- sum_of_column/num_of_rows
print(paste0("The MAPE in percentage for model 3 on the test set is: ", mape_model
3_test))
```

```
## [1] "The MAPE in percentage for model 3 on the test set is: 13.0047411543039"
```

```
print(paste0("The difference between the MAPE value on test and training set for m
odel 3 is ", abs(mape_model3_train - mape_model3_test)))
```

```
## [1] "The difference between the MAPE value on test and training set for model 3
is 0.0157576265536257"
```

From the difference in MAPE value on test and training set I can conclude that for all three models the difference is very small. Model 1 had the largest difference with 1.26 %, followed by model 2 with 0.46 % and finally model 3 with the smallest difference between the test and training values with 0.02 %. When all three models have such low difference I can without bigger harm use any of the three models on other unseen data sets.

But just because a model has a small difference between the training and test set does not mean that the model itself is accurate. For this I can look at the MAPE value. Model 1 has a MAPE value on the test set

for 21.17 %. Model 2 had 12.78 % and model 3 had 13.00 %. Looking at these values the model 2 would be the preferred model to use to forecast values, since it has the smallest size of error in percentage. As a general remark model 2 and model 3 tends to be able to produce the same output on an unseen data set. Both models would be advisable to use, depending on what kind of errors one would like to adjust for. Model 1 seems to perform with a lower accuracy than model 2 and model 3.

Lottery - Stage 4

Viking Lotto

I have chosen one of the Danish lottery systems, known as “Viking Lotto”, which can be found here: https://danskespil.dk/vikinglotto/vindertal (https://danskespil.dk/vikinglotto/vindertal)

The lottery ticket has a game board with 36 elements which are the numbers from 1-36. The player should choose 7 numbers on the game board - this is known as a row.

Once a week the lottery numbers are drawn. There is drawn 7 winning numbers and 2 additional numbers.
The criteria for a success is:
- 6 winning numbers plus 1 of the additional numbers.
- 6 winning numbers.
- 5 winning numbers plus 1 of the additional numbers.
- 5 winning numbers.
- 4 winning numbers plus 1 of the additional numbers.
- 4 winning numbers.
- 3 winning numbers.

In the lottery the sequence of the 7 winning numbers is selected among the first 36 positive integers and the sequence of the winning numbers is not relevant. To calculate the possible sequences of winning numbers I use the formula:
(n,k) = n(n-1)...(n-k+1) / k(k-1)...1
The same formula can be written with factorials:
n! / (k!*(n-k)!), when k <= n, and which is zero when k>n.

```
kombin <- function(n,k)
{
  factorial(n) / (factorial(k)*factorial(n-k))
}
```

The number of possible rows is (36,7) = 8347680.

```
kombin(36,7)
```

```
## [1] 8347680
```

I will start out with the simplest of the cases where I only focus on the winning numbers.
Let x denote the number of correct winning numbers.
The number of rows with x number of correct winning numbers is (7,x)*(29,7-x), which is the number of ways the correct x numbers can be among the correct 7 winning numbers multiplied with the number of ways the remaining 7-x numbers in the row can be selected from the 29 numbers which are not the correct winning numbers.*

P(x) = ((7,x)(29,7-x)) / (36,7)
x = 6

```
(kombin(7,6)*kombin(29,7-6)) / kombin(36,7)
```

```
## [1] 2.431813e-05
```

x = 5

```
(kombin(7,5)*kombin(29,7-5)) / kombin(36,7)
```

```
## [1] 0.001021362
```

x = 4

```
(kombin(7,4)*kombin(29,7-4)) / kombin(36,7)
```

```
## [1] 0.01532042
```

x = 3

```
(kombin(7,3)*kombin(29,7-3)) / kombin(36,7)
```

```
## [1] 0.09958276
```

The above results only focus on the cases where the winning numbers are drawn and not the additional numbers. The additional numbers is two numbers. Let y denote the number of correct additional numbers. I can either have 0, 1 or 2 correct additional numbers.
I will use the following formula:
P(x correct winning numbers and y correct additional numbers) = ((7,x)* (2,y)* (27,7-(x+y))) / (36,7)

First I'll find the number of combinations for x to be among the correct 7 winning numbers, they are multiplied with the number of combinations for y to be among the correct 2 additional numbers, multiplied with the number of ways where the remaining 7-(x and y) numbers of the row is selected among the 27 numbers which is not correct.

If I have **6 winning numbers and 1 additional number** the formula looks like this:
P(6,1) = ((7,6)* (2,1)* (27,7-(6+1))) / (36,7)

```
(kombin(7,6)*kombin(2,1)*kombin(27,7-(6+1))) / kombin(36,7)
```

```
## [1] 1.677113e-06
```

5 winning numbers and 1 additional number

```
(kombin(7,5)*kombin(2,1)*kombin(27,7-(5+1))) / kombin(36,7)
```

```
## [1] 0.0001358461
```

4 winning numbers and 1 additional number

```
(kombin(7,4)*kombin(2,1)*kombin(27,7-(4+1))) / kombin(36,7)
```

```
## [1] 0.002943333
```

It would make sense that the lower the probability the higher the price. The order of the price is related to the number of correct winning numbers and additional numbers: 6 winning numbers + 1 additional number > 6 winning numbers > 5 winning numbers + 1 additional number > 5 winning numbers > 4 winning numbers + 1 additional number > 4 winning numbers > 3 winning numbers.
Let's compare the probability the highest price should have the lowest probability

```
"(kombin(7,6)*kombin(2,1)*kombin(27,7-(6+1))) / kombin(36,7)"<"(kombin(7,6)*kombin
(29,7-6)) / kombin(36,7)<(kombin(7,5)*kombin(2,1)*kombin(27,7-(5+1))) / kombin(36,
7)<(kombin(7,5)*kombin(29,7-5)) / kombin(36,7)<(kombin(7,4)*kombin(2,1)*kombin(27,
7-(4+1))) / kombin(36,7)<(kombin(7,4)*kombin(29,7-4)) / kombin(36,7)<(kombin(7,3)*
kombin(29,7-3)) / kombin(36,7)"
```

```
## [1] TRUE
```

```
(kombin(7,6)*kombin(2,1)*kombin(27,7-(6+1))) / kombin(36,7)
```

```
## [1] 1.677113e-06
```

```
(kombin(7,6)*kombin(29,7-6)) / kombin(36,7)
```

```
## [1] 2.431813e-05
```

```
(kombin(7,5)*kombin(2,1)*kombin(27,7-(5+1))) / kombin(36,7)
```

```
## [1] 0.0001358461
```

```
(kombin(7,5)*kombin(29,7-5)) / kombin(36,7)
```

```
## [1] 0.001021362
```

```
(kombin(7,4)*kombin(2,1)*kombin(27,7-(4+1))) / kombin(36,7)
```

```
## [1] 0.002943333
```

```
(kombin(7,4)*kombin(29,7-4)) / kombin(36,7)
```

```
## [1] 0.01532042
```

```
(kombin(7,3)*kombin(29,7-3)) / kombin(36,7)
```

```
## [1] 0.09958276
```

I have both evaluated that the highest price is associated with the lowest probability and when the probability increases the price is decreasing. It can be observed that there is a lower probability of getting x number of winning numbers and the additional numbers, than just the winning numbers. This makes quite good sense the probability of the correct additional numbers is multiplied in the equation. If we look at the result numbers we have to remember that it is a value between 0-1, where 0 indicates no probability of the outcome (it is impossible) and 1 indicates it will happen for sure. There is a very small chance to win in the lottery, even just to have 3 correct winning numbers.

---

1. From https://www.ncbi.nlm.nih.gov/books/NBK541029/ (https://www.ncbi.nlm.nih.gov/books/NBK541029/), seen 17/12-20.↩
2. From https://www.r-bloggers.com/2020/01/how-to-remove-outliers-in-r/ (https://www.r-bloggers.com/2020/01/how-to-remove-outliers-in-r/), seen 23/12-20.↩
3. From https://canworksmart.com/using-mean-absolute-error-forecast-accuracy/ (https://canworksmart.com/using-mean-absolute-error-forecast-accuracy/), seen 22/12-20.↩
4. From https://canworksmart.com/using-mean-absolute-error-forecast-accuracy/ (https://canworksmart.com/using-mean-absolute-error-forecast-accuracy/), seen 22/12-20.↩