Covariance Matrix & Decision Tree - Qualitfy of Life

Valentin Arbenz, Laszlo Kölliker & Matthias Wenger

November 27, 2020

Contents

1 Introduction

For the hackathon challenge at hand we wanted to get a deeper understand about the survey data at hand.

Our intention in this paper is to use different methods and approaches learned to visualize and fit the data with R.

One of the key-questions that we try to answer with the data is:

"Which factors have a high correlation with the response variable"Lifequalallg" i.e. Life Quality in General?"

2 Preparing the data for analysis

2.1 Loading libraries

```
library(dplyr)
library(mice)
library(tidyr)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(caret)
library(rworldmap)
library(RColorBrewer)
library(randomForest)
library(gbm)
library(corrgram)
```

2.2 Loading the csv-file:

```
d.life_quality <- read.csv2("Data/cleaned_data.csv", encoding = "UTF-8", header = TRUE,
                            na.strings = c("", " ", "-99", "-66", "-77", "NA", "tbd"))
head(d.life_quality)
str(d.life_quality, list.len=ncol(d.life_quality))
```

2.3 Preprocessing & update file:

Deleting survey entries which are not from Lucerne

```
d.life_quality.update <- d.life_quality[d.life_quality$Luzerner==1,]
#drop not relevant columns regarding the correlation matrix
d.life_quality.update <- select(d.life_quality.update, -lfdn, -lastpage, -quality, -duration, -X)
```

Renaming Columns

```
d.life_quality.update.1<-d.life_quality.update %>%
  rename(
    Zf_öv_Arbeitsplatz = v_89,
    Zf_Umgebung_Arbeitsplatz =v_90,
    Zf_Standortattr_Unternehmen =v_336,
    Zf_Standortattr_Innenstadt = v_337,
    Zf_Standortattr_Quartier = v_338,
    Zf_Standortattr_CoWorking = v_364,
    Zf_Wichtigkeit_Läden = v_341,
```

```
    Zf_Gesundheit = v_27,
    Zf_Med_Betreuung = v_294,
    Zf_SicherheitNachts = v_28,
    Zf_SicherheitTag = v_197,
    Zf_SicherheitZuhause = v_101,
    Zf_Wohnsituation1 = v_78,
    Zf_Wohnsituation2 = v_79,
    Zf_ErreichbarkeittInfrast = v_79,
    Zf_PersönlichesEngagement = v_311,
    Zf_ÖVDichte = v_49,
    Zf_ErreichbarkeitÖvWohnung = v_50,
    Zf_VerbindungStadtzentrum = v_51,
    Zf_VerbindungNaherholung= v_54,
    Zf_VerbindungArbeitsplatz= v_53,
    Zf_MobilitätGrünphase = v_187,
    Zf_MobilitätSitzbankDichte = v_188,
    Zf_Veloparkplätze =v_279,
    Zf_Freizeit = v_55,
    Zf_Kultur =v_56,
    Zf_SicherheitÖv = v_39,
    Zf_SicherheitAutoMotorrad = v_40,
    Zf_SicherheitVelo = v_41,
    Zf_SicherheitFuss = v_42,
    Zf_Velo_AnbindungÖv = v_323,
    Zf_AnbindungVelo = v_324,
    Zf_Bus_AnbindungVelo = v_325,
    Zf_Vertrauen_StadtVerw = v_126,
    Zf_Einbringung = v_366,
    Zf_Einbringung2 = v_367,
    Zf_Sorgen_Alter1 = v_256,
    Zf_Sorgen_Alter2 = v_257,
    Zf_Sorgen_Alter3 = v_258,
    Zf_Sorgen_Alter4 = v_259,
    Zf_Sorgen_Alter5 = v_260,
    Zf_Sorgen_Umwelt1 = v_317,
    Zf_Sorgen_Umwelt2 = v_318,
    Zf_Sorgen_Umwelt3 = v_319,
    Zf_Sorgen_Umwelt4 = v_320,
    Zf_Sorgen_Umwelt5 = v_321,
    Zf_Lebensqualität = Lebensqualallg,
    Zf_Kinder1 =  v_73,
    Zf_Kinder2 = v_74,
    Zf_Kinder3 = v_75,
    Zf_Kinderbetr1 = v_152,
    Zf_Kinderbetr2 = v_153,
    Zf_Kinderbetr3 = v_154,
    Zf_FamilieBeruf1 = v_269,
    Zf_FamilieBeruf2 = v_274,
    Zf_FamilieBeruf3 = v_275
  )
```

Dropping columns with the regEx "v_"

```
d.life_quality.update.2 <- d.life_quality.update.1[, -grep("v_", colnames(d.life_quality.update.1))]
```

Checking the structure:

```
str(d.life_quality.update.2, list.len=ncol(d.life_quality.update.2))
```

```
## 'data.frame':    630 obs. of  66 variables:
##  $ Luzerner                    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Sex                         : int  1 2 1 2 1 2 2 2 1 1 ...
##  $ Altergruppe                 : int  1 3 3 3 3 3 3 6 3 4 ...
##  $ Beziehungstatus             : int  1 2 2 2 2 1 1 2 2 1 ...
##  $ Quartier                    : int  5 1 5 1 1 1 1 1 1 6 ...
##  $ Zuzug                       : int  2 1 1 2 2 6 2 5 2 5 ...
##  $ Bildung                     : int  8 7 7 7 7 7 4 3 7 7 ...
##  $ HH                          : int  4 3 1 5 5 5 5 2 4 4 ...
##  $ Kinder_Schule               : int  1 2 2 2 2 2 2 2 2 1 ...
##  $ Erwerb                      : int  3 5 1 1 1 5 3 2 1 1 ...
##  $ Zf_Lebensqualität           : int  5 6 6 5 6 5 5 5 5 5 ...
##  $ ArbeitsplatzLuzern          : int  NA NA 2 2 1 NA NA NA 2 2 ...
##  $ Zf_Umgebung_Arbeitsplatz    : int  NA NA NA NA 2 NA NA NA NA NA ...
##  $ Zf_Standortattr_Unternehmen: int  2 2 2 2 2 5 3 2 3 3 ...
##  $ Zf_Standortattr_Innenstadt : int  2 1 1 2 2 2 3 1 2 2 ...
##  $ Zf_Standortattr_Quartier   : int  1 1 2 2 2 1 3 2 2 2 ...
##  $ Zf_Standortattr_CoWorking  : int  2 5 2 5 3 5 5 1 2 5 ...
##  $ Zf_Wichtigkeit_Läden       : int  2 1 2 2 2 2 2 1 2 3 ...
##  $ Zf_Kinder1                 : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_Kinder2                 : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_Kinder3                 : int  2 NA NA NA NA NA NA NA NA 3 ...
##  $ Zf_Kinderbetr1             : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_Kinderbetr2             : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_Kinderbetr3             : int  2 NA NA NA NA NA NA NA NA 5 ...
##  $ Zf_FamilieBeruf1           : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_FamilieBeruf2           : int  2 NA NA NA NA NA NA NA NA 2 ...
##  $ Zf_FamilieBeruf3           : int  2 NA NA NA NA NA NA NA NA 1 ...
##  $ Zf_Gesundheit              : int  2 6 6 2 1 1 1 1 1 1 ...
##  $ Zf_Med_Betreuung           : int  2 5 5 5 5 5 5 1 5 1 ...
##  $ Zf_SicherheitNachts        : int  2 2 1 2 1 1 2 3 1 1 ...
##  $ Zf_SicherheitTag           : int  2 2 1 1 1 1 1 3 1 1 ...
##  $ Zf_SicherheitZuhause       : int  2 1 1 1 1 1 1 1 1 1 ...
##  $ Zf_Wohnsituation1          : int  2 1 1 2 2 2 1 1 1 2 ...
##  $ Zf_ErreichbarkeittInfrast  : int  2 2 1 2 1 1 1 1 1 2 ...
##  $ Zf_Freizeit                : int  2 1 2 2 1 1 2 3 1 2 ...
##  $ Zf_Kultur                  : int  3 2 2 1 1 2 2 1 2 2 ...
##  $ Zf_PersönlichesEngagement  : int  2 2 3 4 3 4 4 1 4 2 ...
##  $ Zf_SicherheitÖv            : int  2 2 1 2 1 1 1 2 2 2 ...
##  $ Zf_SicherheitAutoMotorrad  : int  2 1 1 5 2 1 5 1 2 2 ...
##  $ Zf_SicherheitVelo          : int  2 1 1 2 3 2 3 3 3 2 ...
##  $ Zf_SicherheitFuss          : int  2 1 1 1 1 2 1 2 1 1 ...
##  $ Zf_ÖVDichte                : int  2 1 1 2 2 1 2 5 1 1 ...
##  $ Zf_ErreichbarkeitÖvWohnung : int  2 1 1 3 2 1 1 1 1 1 ...
##  $ Zf_VerbindungStadtzentrum  : int  2 1 1 5 1 5 5 1 1 1 ...
##  $ Zf_VerbindungNaherholung   : int  2 1 1 2 1 1 2 1 1 1 ...
##  $ Zf_VerbindungArbeitsplatz  : int  2 1 3 2 1 1 2 4 1 2 ...
```

```
##  $ Zf_MobilitätGrünphase      : int  2 1 1 2 1 2 1 1 2 1 ...
##  $ Zf_MobilitätSitzbankDichte : int  2 1 1 2 2 1 2 2 1 1 ...
##  $ Zf_Veloparkplätze          : int  2 2 3 5 3 3 5 1 3 3 ...
##  $ Zf_Velo_AnbindungÖv        : int  2 2 2 5 2 5 5 2 5 2 ...
##  $ Zf_AnbindungVelo           : int  2 1 2 5 2 5 5 5 5 2 ...
##  $ Zf_Bus_AnbindungVelo       : int  2 1 2 5 2 5 5 5 5 2 ...
##  $ Zf_Vertrauen_StadtVerw     : int  2 5 5 2 1 2 2 5 2 2 ...
##  $ Zf_Einbringung             : int  3 5 5 5 2 5 2 3 5 2 ...
##  $ Zf_Einbringung2            : int  4 5 5 5 2 5 2 2 5 2 ...
##  $ Zf_Sorgen_Umwelt1          : int  1 2 2 2 2 1 2 1 1 2 ...
##  $ Zf_Sorgen_Umwelt2          : int  5 2 2 2 2 2 2 1 2 2 ...
##  $ Zf_Sorgen_Umwelt3          : int  1 3 3 2 2 1 2 1 2 2 ...
##  $ Zf_Sorgen_Umwelt4          : int  4 3 3 2 3 2 3 1 3 3 ...
##  $ Zf_Sorgen_Umwelt5          : int  2 3 1 2 2 2 3 1 2 2 ...
##  $ Zf_Sorgen_Alter1           : int  3 3 2 2 3 3 2 3 2 3 ...
##  $ Zf_Sorgen_Alter2           : int  4 3 3 2 3 3 3 3 4 3 ...
##  $ Zf_Sorgen_Alter3           : int  4 3 2 2 2 3 3 2 3 2 ...
##  $ Zf_Sorgen_Alter4           : int  3 3 1 2 2 2 3 1 3 2 ...
##  $ Zf_Sorgen_Alter5           : int  4 5 0 0 4 5 0 3 0 0 ...
##  $ Corona                     : int  2 2 2 1 1 1 2 1 1 2 ...
```

2.4 Exploring NA patterns

```r
md.pattern(d.life_quality.update.2, plot = FALSE)
```

Using the md.pattern command from the mice-package, we can see how much missing values we have in the data.

Since we don‘t want NA in the data for our models, we have to get rid of them. Replacing the NA (for example with the mean) does not make much sense here, so we decided to delete the columns with the missing values.

Note that we decided to do this, as we do not want to na.omit() the data as this would reduce the size of the data to 1/3 of its original size. Additionally, most often the NAs relate to a question which was not answered. This also includes additional questions based on the initial question.

2.5 Drop missing value columns

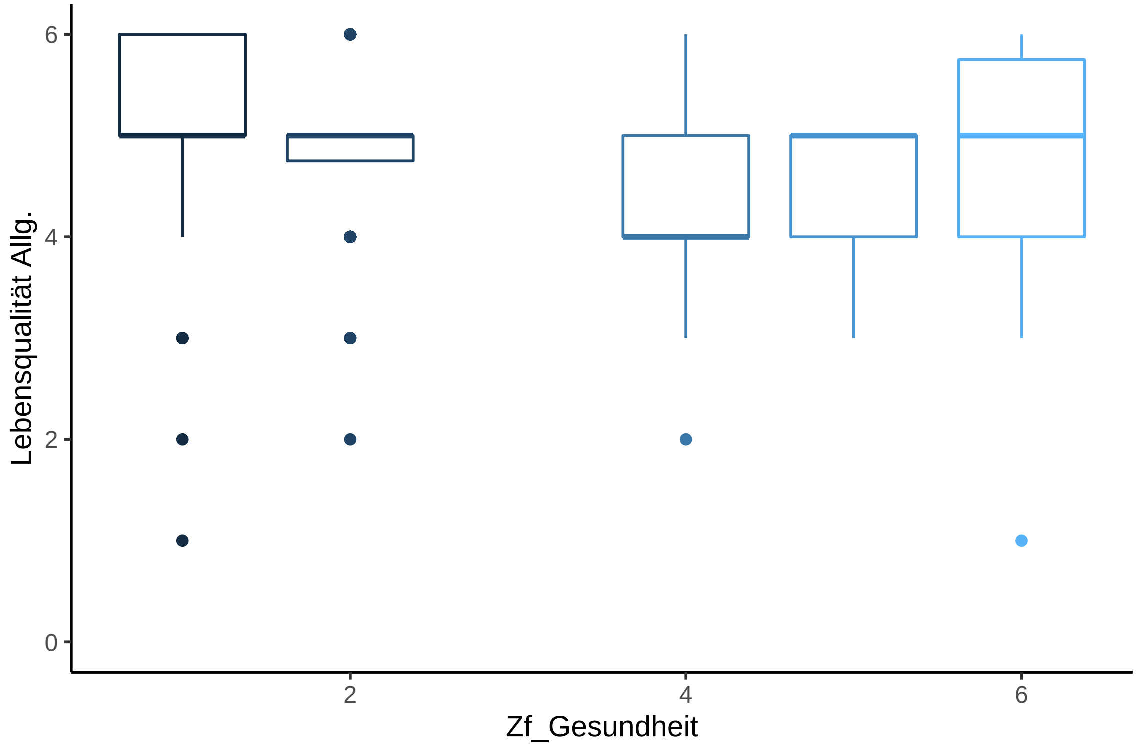
```r
test_data <- select(d.life_quality.update.2, -ArbeitsplatzLuzern, -Zf_Umgebung_Arbeitsplatz, -Zf_Kinder

md.pattern(test_data, plot=FALSE)
```

After dropping the NA we are now left with a cleaned data-frame containing 630 observations.

3 Visualizing the data (examples)

3.1 Lebensqualität Allg. vs. Gesundheit

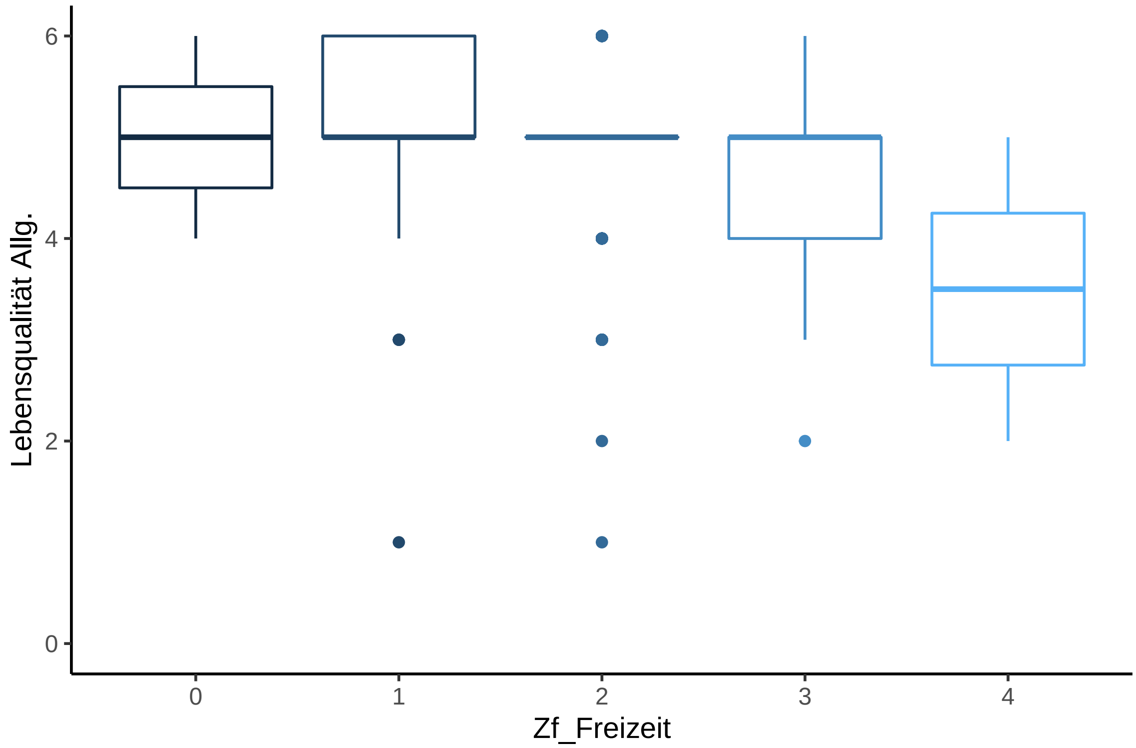
```
ggplot(data = test_data, aes(x = Zf_Gesundheit, y = Zf_Lebensqualität, color = Zf_Gesundheit, group = Zf
  geom_boxplot() +
  theme_minimal() +
  theme_classic() +
  theme(legend.position = "none") +
  scale_y_continuous(name = "Lebensqualität Allg.", limits = c(0,NA))
```



In the first plot wee already see some interesting insights. The scale is inverse - meaning the answers rank from 1 to 6, with 1 being the "best" answer. It seems that the "worse" the Zufriedenheit in Gesundheit is (meaning 4+ Score), the higher the chance the Lebensqualität Allg. Score drops.

3.2 Lebensqualität Allg. vs. Freizeit

```
ggplot(data = test_data, aes(x = Zf_Freizeit, y = Zf_Lebensqualität, color = Zf_Freizeit, group = Zf_Fre
  geom_boxplot() +
  theme_minimal() +
  theme_classic() +
  theme(legend.position = "none") +
  scale_y_continuous(name = "Lebensqualität Allg.", limits = c(0,NA))
```



We also get a certain picture in regards to the Freizeit vs. Lebensqualität Allg. It seems that the less happy the people were with the "Freizeit" topic, the higher the chance the Lebensqualität Allg. Score is reduced.

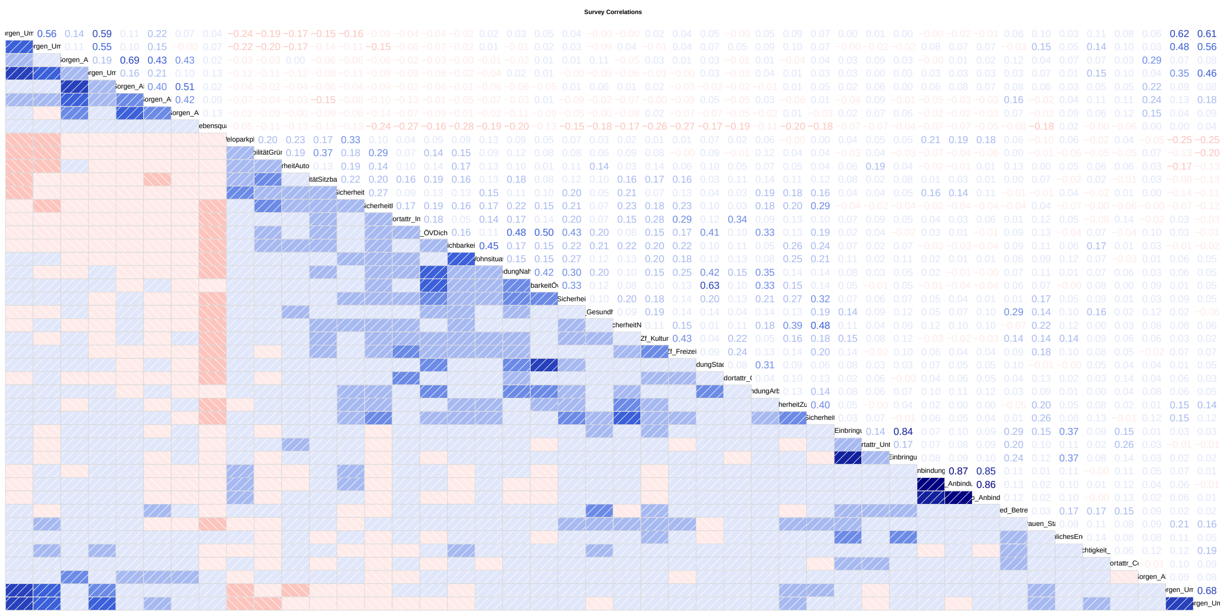
4 Fitting models

4.1 Correlation Matrix

To start really simple we fit a correlation matrix to get further insights.

Attention: We must not forget, that the response variable "ZF_Lebensqualität" correlates negatively, as the rating from the questions is inverse (meaning: 1 is the best, 6 the worst possible answer)

```
test_used <- test_data[,grep("Zf", colnames(test_data))]

corrgram_life_quality <- corrgram <- corrgram(test_used, order=TRUE,
         upper.panel=panel.cor, main="Survey Correlations",
         cex.labels = 2, label.pos = c(0.5, 0.5)
         )
```



With a first look we see that some variables correlate higher in negative way as others with the response variable "Lebensqualität Allg.", which indicates a higher influence on the response variable. In a next step we will primarily focus on the more relevant factors:

```
final_data <- select(test_data, Zf_Lebensqualität, Zf_ErreichbarkeittInfrast, Zf_Freizeit, Zf_Standorta

corrgram <- corrgram(final_data, order=TRUE,
         upper.panel=panel.cor, main="Survey Correlations",
         cex.labels = 2.5
         )
```

This correlation matrix showcases that 12 of all the survey factors correlate in a high manner with the "ZF_Lebensqualität". The five most impactful variables are:

1) Zf_ErreichbarkeitInfrastruktur
2) Zf_Freizeit
3) Zf_Standortattr_Innenstadt
4) Zf_Kultur
5) Zf_SicherheitFuss

4.2 Fitting a simple linear model:

```r
lm.life_quality <- lm(Zf_Lebensqualität ~ .,
               data = final_data)
summary(lm.life_quality)
```

```
##
## Call:
## lm(formula = Zf_Lebensqualität ~ ., data = final_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5685 -0.3133  0.0262  0.4002  3.1563
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 6.538586   0.131734  49.635  < 2e-16 ***
## Zf_ErreichbarkeittInfrast  -0.207329   0.049121  -4.221  2.8e-05 ***
## Zf_Freizeit                -0.100755   0.049526  -2.034 0.042338 *
## Zf_Standortattr_Innenstadt -0.127537   0.036448  -3.499 0.000500 ***
## Zf_Kultur                  -0.097084   0.047998  -2.023 0.043536 *
## Zf_SicherheitFuss          -0.152309   0.045940  -3.315 0.000969 ***
## Zf_SicherheitZuhause       -0.122168   0.073279  -1.667 0.095992 .
## Zf_VerbindungNaherholung   -0.012074   0.035770  -0.338 0.735823
## Zf_Standortattr_Quartier   -0.056183   0.031860  -1.763 0.078322 .
```

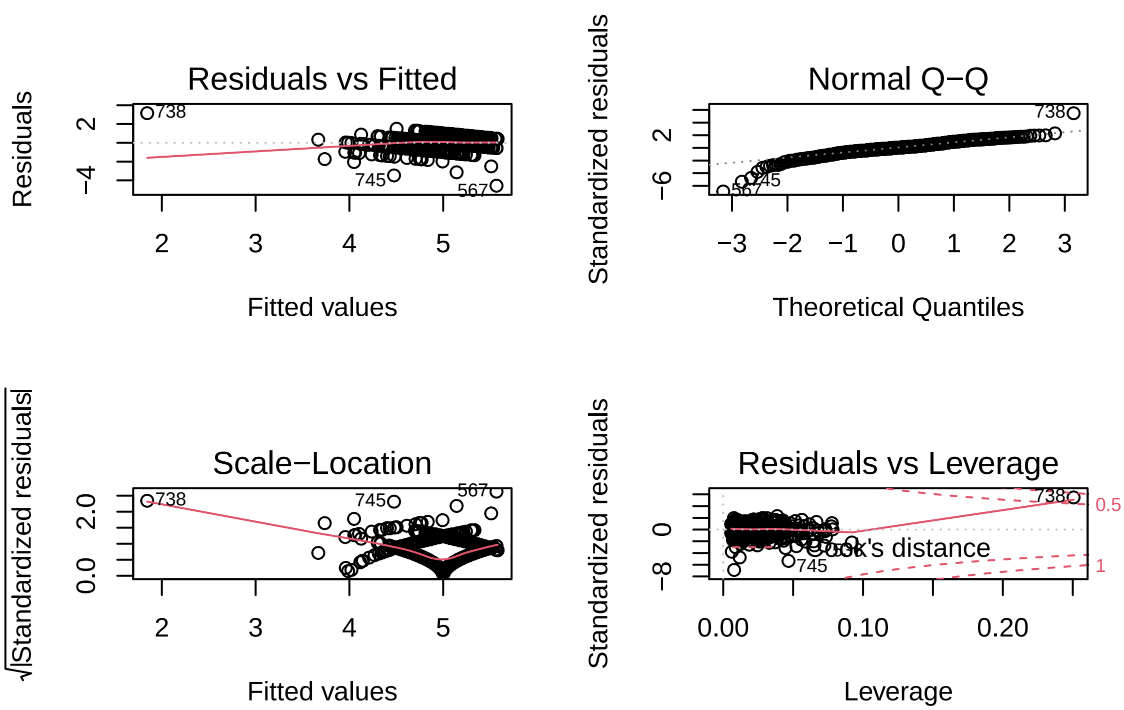
```
## Zf_Wohnsituation1           0.011539   0.050049   0.231 0.817743
## Zf_Gesundheit              -0.042345   0.020744  -2.041 0.041640 *
## Zf_SicherheitTag            0.006151   0.079484   0.077 0.938346
## Zf_VerbindungStadtzentrum  -0.069974   0.031173  -2.245 0.025141 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6655 on 617 degrees of freedom
## Multiple R-squared:  0.2205, Adjusted R-squared:  0.2054
## F-statistic: 14.55 on 12 and 617 DF,  p-value: < 2.2e-16
```

Looking at the summary of the linear model, the Zf_ErreichbarkeittInfrast, Zf_Standortattr_Innenstadt & Zf_SicherheitFuss score both have a significant effect on the response variable.

The adjusted R-squared is 0.2054, so about 20% of the variation is described by the model.

4.3 Examining the model diagnostics:

```r
par(mfrow=c(2,2))
plot(lm.life_quality)
```



The assumption of normal errors with constant variance does seem to be not fulfilled ("homoscedasticity assumption").

On plot number two (i.e. the Quantile-Quantile plot) there seem to be a slight deviation from the expected line before the -2 quantile.

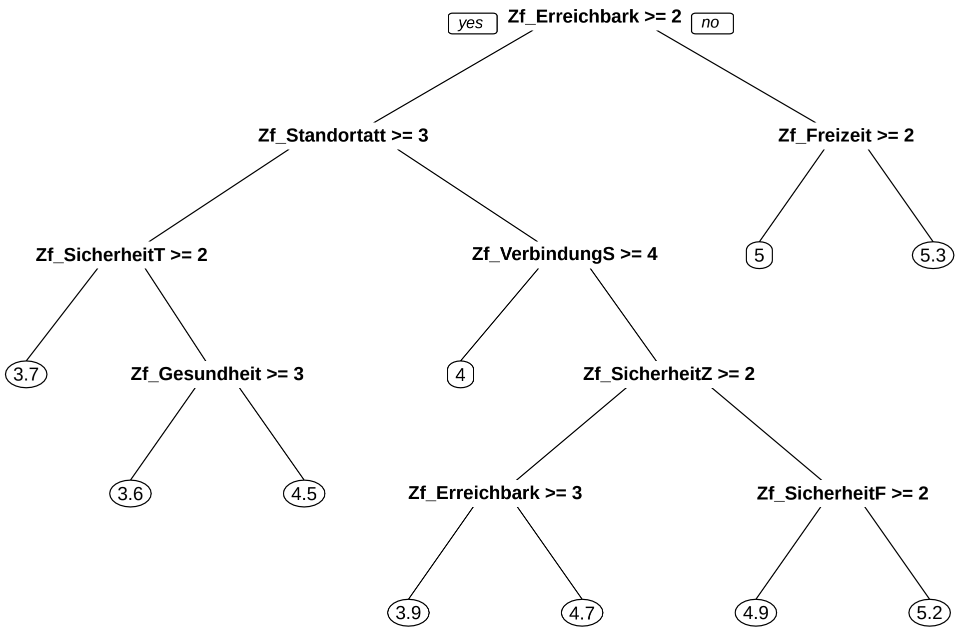
Plot number three (i.e. the Scale-Location plot) indicates that the variance of the residuals increase with the fitted values. Therefore, we can conclude that the assumptions of this Linear Model are not perfectly fulfilled.

5 Fit further models

5.1 Regression Tree

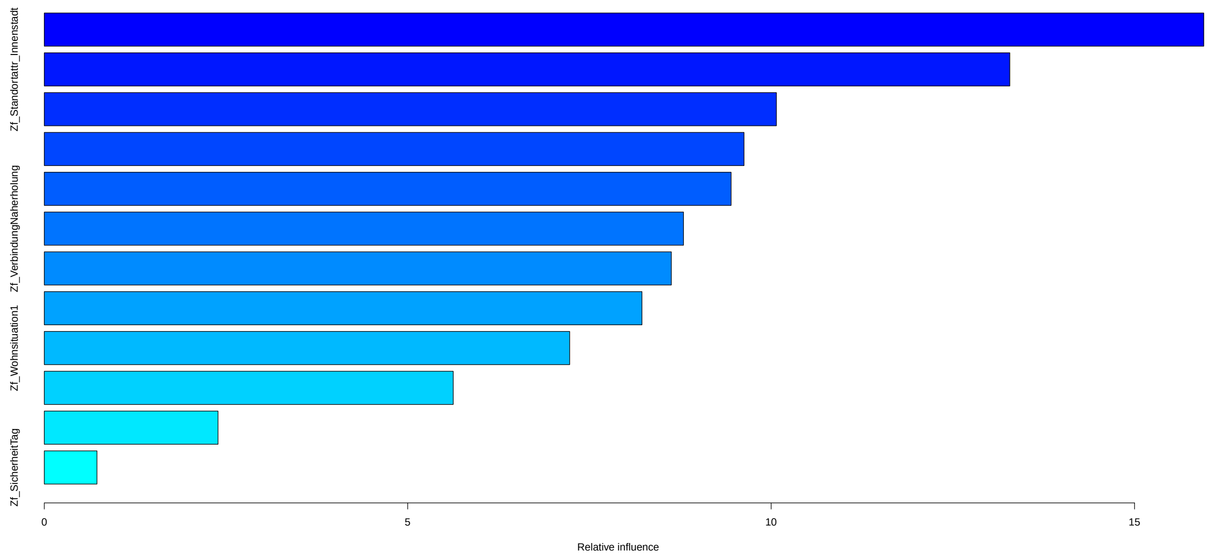
For the second model we decided to fit a regression tree. For this we used the rpart-package.

```
tree.life_quality <- rpart(Zf_Lebensqualität ~ .,
               data = final_data, method = "anova", control=rpart.control(minsplit = 16))
prp(tree.life_quality, cex = 0.6)
```



We decided to implement the minsplit parameter to 16 to also include the ZF_Gesundheit aspect. The key-split node is described as "Zf_ErreichbarkeittInfrastruktur".

```
gbm.life_quality <- gbm(Zf_Lebensqualität ~ .,
               data = final_data,
                 distribution="gaussian",n.trees=5000, interaction.depth=12)
summary(gbm.life_quality)
```




```
##                                                     var   rel.inf
## Zf_Standortattr_Quartier     Zf_Standortattr_Quartier 15.952482
## Zf_Standortattr_Innenstadt Zf_Standortattr_Innenstadt 13.283905
## Zf_ErreichbarkeittInfrast   Zf_ErreichbarkeittInfrast 10.072343
## Zf_SicherheitFuss                   Zf_SicherheitFuss  9.627300
## Zf_Freizeit                               Zf_Freizeit  9.449476
## Zf_VerbindungNaherholung     Zf_VerbindungNaherholung  8.794322
## Zf_Gesundheit                           Zf_Gesundheit  8.626729
## Zf_Kultur                                   Zf_Kultur  8.223158
## Zf_Wohnsituation1                   Zf_Wohnsituation1  7.228677
## Zf_VerbindungStadtzentrum   Zf_VerbindungStadtzentrum  5.626280
## Zf_SicherheitZuhause             Zf_SicherheitZuhause  2.390380
## Zf_SicherheitTag                     Zf_SicherheitTag  0.724948
```

The boosted model indicates also that Zf_Standortattr_Quartier, Zf_Standortattr_Innenstadt, Zf_SicherheitFuss, Zf_ErreichbarkeittInfrast, Zf_Freizeit have a higher relative influence.

```r
rF.life_quality <- randomForest(Zf_Lebensqualität ~ .,
               data = final_data)
print(rF.life_quality) # view results
```

```
##
## Call:
##  randomForest(formula = Zf_Lebensqualität ~ ., data = final_data)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 0.4765915
##                     % Var explained: 14.35
```

```r
importance(rF.life_quality) # importance of each predictor
```

```
##                            IncNodePurity
## Zf_ErreichbarkeittInfrast       27.92097
## Zf_Freizeit                     19.03535
```

```
## Zf_Standortattr_Innenstadt      29.81633
## Zf_Kultur                       20.71555
## Zf_SicherheitFuss               18.37233
## Zf_SicherheitZuhause            12.84063
## Zf_VerbindungNaherholung        16.62395
## Zf_Standortattr_Quartier        25.47181
## Zf_Wohnsituation1               15.32932
## Zf_Gesundheit                   23.78685
## Zf_SicherheitTag                 8.56851
## Zf_VerbindungStadtzentrum       26.33910
```

The randomForest model indicates that Zf_Standortattr_Innenstadt, Zf_ErreichbarkeittInfrast, Zf_VerbindungStadtzentrum , Zf_Standortattr_Quartier, Zf_Gesundheit have a higher impact on the response variable.

6 Comparing the different models

6.1 Crossvalidation lm

We use a 10-fold crossvalidation to see how good the linear model performed. We do this with the caret-package.

```
set.seed(1)
train.control.lm <- trainControl(method = "cv", number = 10)
cv.lm <- train(Zf_Lebensqualität ~ .,
               data = final_data,
               method = "lm",
               trControl = train.control.lm)
print(cv.lm)
```

```
## Linear Regression
##
## 630 samples
##  12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 568, 567, 567, 567, 567, 566, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.6649925  0.2060907  0.4815497
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

With 10-fold cross validation we get a RMSE (Residual Mean Squared Error) of 0.6649 for the linear model. This means that on average the predictions of our linear model deviate from the observations by about 0.6649 score points.

6.2 Crossvalidation tree

The same function is applied for the tree model.

```
set.seed(1)
train.control.tree <- trainControl(method = "cv", number = 10)
cv.tree <- train(Zf_Lebensqualität ~ .,
               data = final_data,
                 method = "rpart",
                 trControl = train.control.tree)
print(cv.tree)
```

```
## CART
##
## 630 samples
##  12 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 568, 567, 567, 567, 567, 566, ...
## Resampling results across tuning parameters:
##
##   cp          RMSE       Rsquared    MAE
##   0.02863918  0.6985636  0.13296202  0.4804271
```

```
##   0.05610084  0.7212873  0.06457724  0.5071985
##   0.07185547  0.7437321  0.00452120  0.4777866
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was cp = 0.02863918.
```

For the regression tree we get 0.6985 as the best possible RMSE.

6.3 Results

LM: RSME 0.6649925

Tree: RSME 0.6985636

It seems that the linear model has a slightly better performance that the regression tree.

7 Conclusion

Having had the opportunity to work with the dataset provided in the context of "Qualitfy of Life in Lucerne" we gained interesting insights in what aspects have a higher impact on the overall "Life Quality Zufriedenheit".

After loading & cleaning (NAs) the data set, we used a linear model (lm) and a regression tree (tree) to answer our hypothesis: **"Which variables have a high correlation with the response variable"Lebensqualität Allg."?"**

As different models sometimes slightly deviate from others, it is nice to see that some patterns emerge:

1) Zf_ErreichbarkeittInfrast has in all models a higher relative impact and could be use as a 1st split-node.

2) Zf_Standortattr_Innenstadt & Zf_Standortattr_Quartier could be used as a direct follow up to the 1st split-node regarding relative relevance.

3) Most often in a 3rd split, the impact is shared between Zf_Gesundheit, Zf_Freizeit

The 10-fold crossvalidation concluded:

LM: RSME 0.6649925

Tree: RSME 0.6985636

Meaning, that the linear model has a slightly (rather minimal) better performance that the regression tree. So if we would do a prediction with given predictors, the response variable could deviate (mean) around ~ 0.6649 score points from the true value.