Homework

UK Hochdimensionale & Komplexe Statistik

Cordula Eggerth

Wintersemester 2018/19

Task 1:

Get acquainted with the dataset DataCar in R.

https://cran.r-project.org/web/packages/insuranceData/insuranceData.pdf

The dataset DataCar contains data on 1-year vehicle insurance policies (initiated in 2004/05). Installing the package `insuranceData` is a prerequisite to use the dataset. The dataframe comprises 67856 rows (i.e. observations) and 11 columns (i.e. variables).

```
> head(dataCar, n=10)
   veh_value  exposure clm numclaims claimcst0 veh_body veh_age gender area agecat          X_OBSTAT_
1       1.06 0.3039014   0         0         0    HBACK       3      F    C      2 01101    0    0    0
2       1.03 0.6488706   0         0         0    HBACK       2      F    A      4 01101    0    0    0
3       3.26 0.5694730   0         0         0      UTE       2      F    E      2 01101    0    0    0
4       4.14 0.3175907   0         0         0    STNWG       2      F    D      2 01101    0    0    0
5       0.72 0.6488706   0         0         0    HBACK       4      F    C      2 01101    0    0    0
6       2.01 0.8542094   0         0         0    HDTOP       3      M    C      4 01101    0    0    0
7       1.60 0.8542094   0         0         0    PANVN       3      M    A      4 01101    0    0    0
8       1.47 0.5557837   0         0         0    HBACK       2      M    B      6 01101    0    0    0
9       0.52 0.3613963   0         0         0    HBACK       4      F    A      3 01101    0    0    0
10      0.38 0.5201916   0         0         0    HBACK       4      F    B      4 01101    0    0    0
```

`veh_value` … value of vehicle in ten-thousand USD

`exposure` … value ∈ [0,1]

`clm` ... boolean variable whether claim occurred or not

`numclaims` … number of claims (per observation)

`claimcst0` … amount per claim (in case of no claim: 0)

`veh_body` … vehicle body (coded as the respective abbreviation)

`veh_age` … category 1,2,3,4 (with 1 being the youngest)

`gender` … F (female), M (male)

`area` … 1,2,3,4,5,6

`agecat` … category 1,2,3,4,5,6 (with 1 being the youngest)

`X_OBSTAT_` … factor (levels: 01101 0 0 0

Describe (mathematically) how Poisson regression can be used to fit count data.[1]

Commonly, the following assumptions are made for Poisson Models:

- Independent observations
- Homogeneous distribution of events of interest
- Specified time

According to Poisson distribution, E[X]=Var[X]=λ in a theoretical sense. From a practical viewpoint, this equality does, however, often not hold, due to overdispersion (as each type of event does not have the same probability of occurrence and the variance may in practice be larger than theory would suggest).

Count data, as prevalent in this case, refers to non-negative integers that denote the number of occurrences of events (here: insurance claims) within a fixed time. In insurance data, few claim events amount to a very high sum, and the majority of events involve a low claim amount.

By using the Poisson regression model, such insurance count data should be modelled as a function of variables. For the Poisson distribution, we assume events to occur at random and uniformly in time.

Poisson distribution: $P(x) = \frac{\lambda^x e^{-\lambda}}{x!}$ with x = 0,1,2… .

Given the Poisson distribution, we can do Poisson Regression:

$$P(Y_i = j \mid X_i) = \frac{\lambda_i^j e^{-\lambda_i}}{j!} \text{ with j=0,1,2… .}$$

$$\lambda_i = E[Y_i|X_i] = Var[Y_i|X_i] = \exp(X_i'\beta), \text{ wobei } \lambda_i > 0.$$

Indicate the model as:

$$\lambda_i = \exp(X_i'\beta) > 0$$

Formulate (the non-linear regression model with heteroscedastic variances):

$$y_i = \exp(X_i'\beta) + \varepsilon_i$$

Use Maximum-Likelihood and compute log-likelihood:

$$\log\big(L(\beta)\big) = \sum_{i=1}^{N} y_i X_i{}'\,\beta + \exp(X_i'\beta) - \ln(y_i!)$$

First Order Conditions (of regression model):

$$\frac{\delta Log(L(\beta))}{\delta\beta'} = \sum_{i=1}^{N}[y_i - \exp(X_i'\beta)]X_i = 0$$

Second Order Conditions (of regression model):

[1] Sources (refer to the whole chapter): https://newonlinecourses.science.psu.edu/stat504/node/169/; https://pareonline.net/getvn.asp?v=21&n=2; https://cran.r-project.org/web/packages/pscl/vignettes/countreg.pdf.

$$\frac{\delta^2 Log(L(\beta))}{\delta\beta\ \delta\beta'} = \sum_{i=1}^{N}[-\exp(X_i'\beta)]X_i X_i'$$

According to Maximum-Likelihood theory, β_{ML} is asymptotically normal with mean β. The variance is: $Var[\beta\ |X] = (\ \sum_{i=1}^{N}[\exp(X_i'\beta)]X_i X_i'\ )^{-1}$.

Equidispersion holds if: $E[Y_i|X_i] = Var[Y_i|X_i]$. If this is not the case, Poisson distribution is not the adequate model to fit the data. In case of overdispersion, the observed variance is larger than the variance assumed by the theoretical model. To deal with the problem of overdispersion, a specific dispersion parameter can be used (in line with a Quasi-Poisson model), or the distribution can be changed to negative binomial.

In the regression model, the outcome variable Y is a count of events.

The regressors X = $(X_1, \dots, X_k)$ can be continuous and/or categorical variables.

Write the model as a General Linear Model (GLM) for counts:

$$g(\lambda) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k = x_i'\beta$$

For the regression, e.g. identity link or natural log link can be used.

The identity link model $\lambda = \beta_0 + \beta_1 x_1$ could also lead to negative values for λ. Therefore, we consider the natural log link model (which is also the default case in the used R libraries[2]) log(λ) = $\beta_0 + \beta_1 x_1$. This is equivalent to: $\lambda = \exp(\beta_0 + \beta_1 x_1) = \exp(\beta_0)\exp(\beta_1 x_1)$.

$\exp(\beta_0)$ … effect on mean(Y)

$\exp(\beta_1)$ … for each increase of X by 1 unit, there is a multiplicative effect on mean(Y).

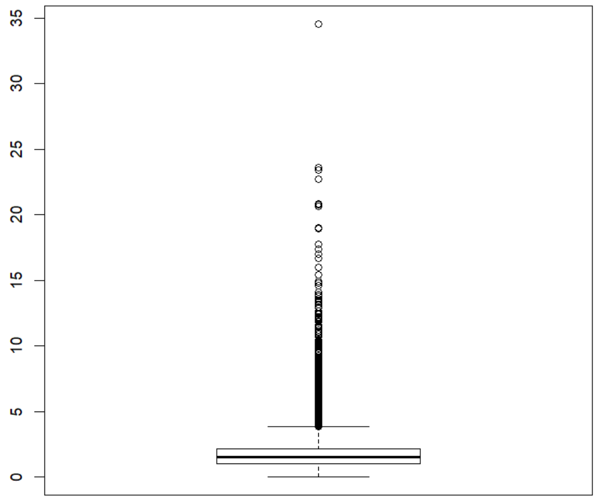
[2] And yields only non-negative values.

For the dataset DataCar use R to display descriptive statistics of interest about the number and distributions of claims (also across different areas, vehicle ages, etc.).

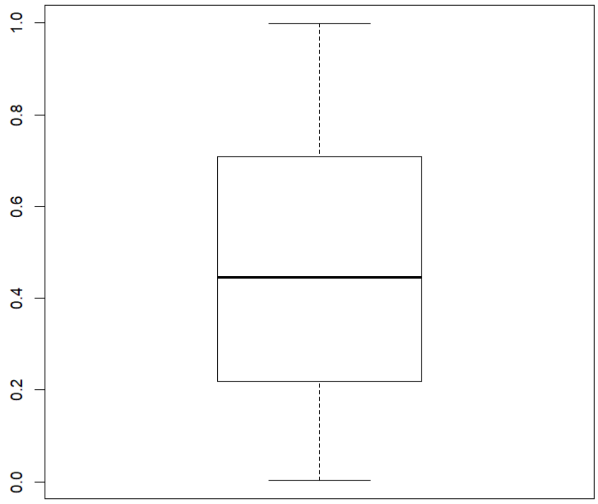
Summary statistics on the dataset:

```
> summary(dataCar) # min, 1st quantile, median, mean, 3rd quantile, max
   veh_value          exposure             clm              numclaims          claimcst0            veh_body        veh_age       gender
 Min.   : 0.000   Min.   :0.002738   Min.   :0.00000   Min.   :0.00000   Min.   :    0.0   SEDAN  :22233   Min.   :1.000   F:38603
 1st Qu.: 1.010   1st Qu.:0.219028   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:    0.0   HBACK  :18915   1st Qu.:2.000   M:29253
 Median : 1.500   Median :0.446270   Median :0.00000   Median :0.00000   Median :    0.0   STNWG  :16261   Median :3.000
 Mean   : 1.777   Mean   :0.468651   Mean   :0.06814   Mean   :0.07276   Mean   :  137.3   UTE    : 4586   Mean   :2.674
 3rd Qu.: 2.150   3rd Qu.:0.709103   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:    0.0   TRUCK  : 1750   3rd Qu.:4.000
 Max.   :34.560   Max.   :0.999316   Max.   :1.00000   Max.   :4.00000   Max.   :55922.1   HDTOP  : 1579   Max.   :4.000
                                                                                           (Other): 2532
 area          agecat                     X_OBSTAT_
 A:16312   Min.   :1.000   01101    0    0    0:67856
 B:13341   1st Qu.:2.000
 C:20540   Median :3.000
 D: 8173   Mean   :3.485
 E: 5912   3rd Qu.:5.000
 F: 3578   Max.   :6.000
```

Boxplot of `veh_value`:



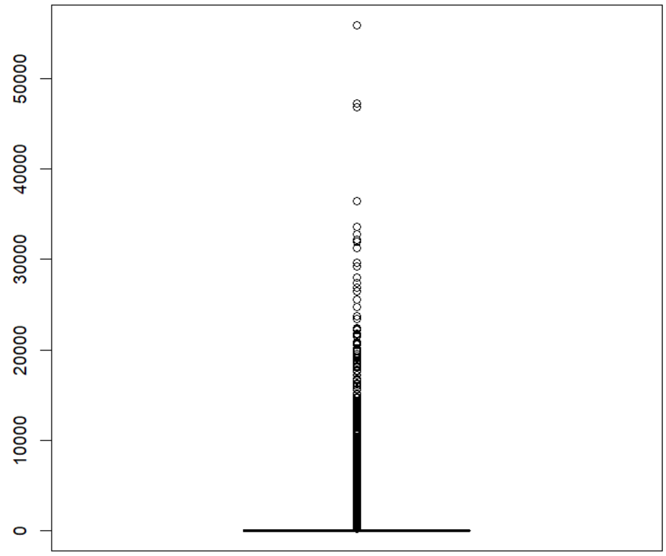
Boxplot of `exposure`:



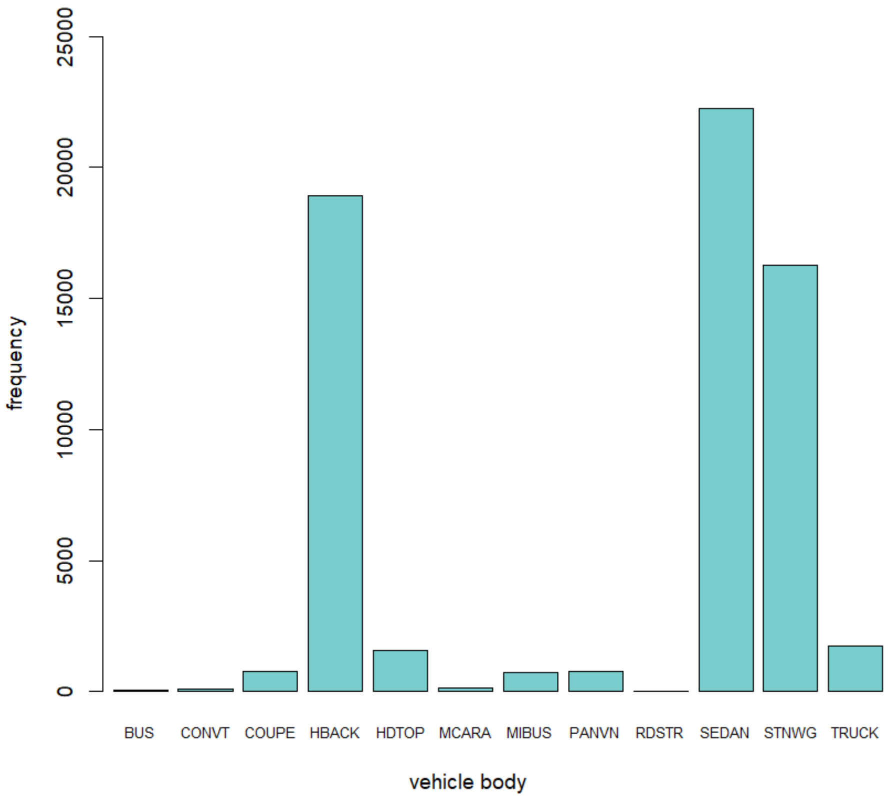
Proportion of claims (events) occurring based on all observations:

```
> sum(dataCar$clm)/length(dataCar$clm)
[1] 0.06814431
```

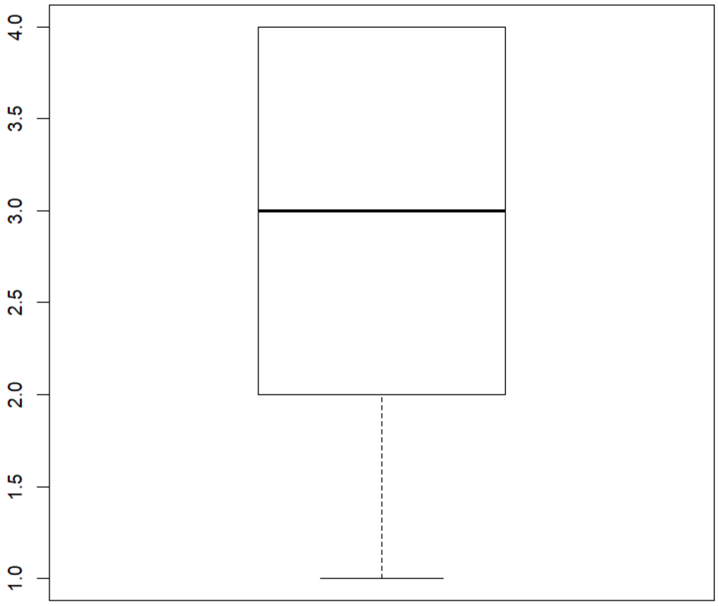
Boxplot `claimcst0` (amount of claim):



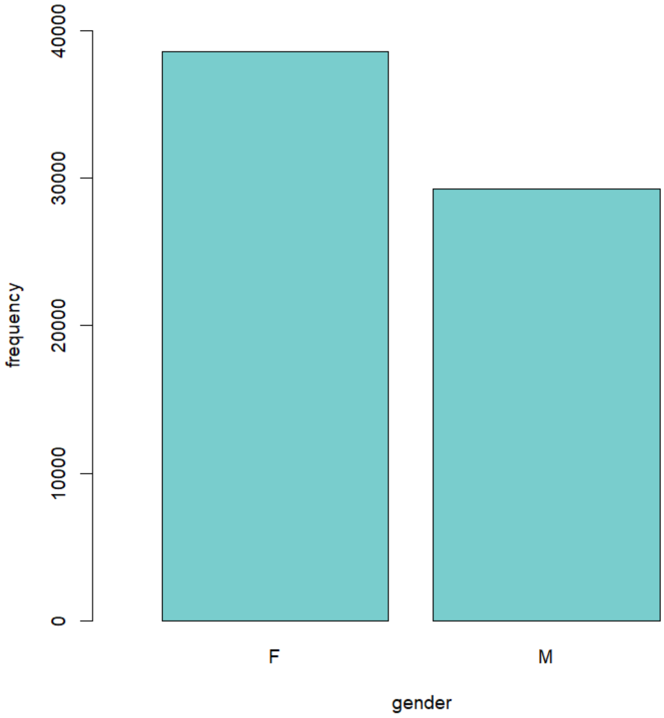
Distribution of `veh_body` (within the whole dataset):



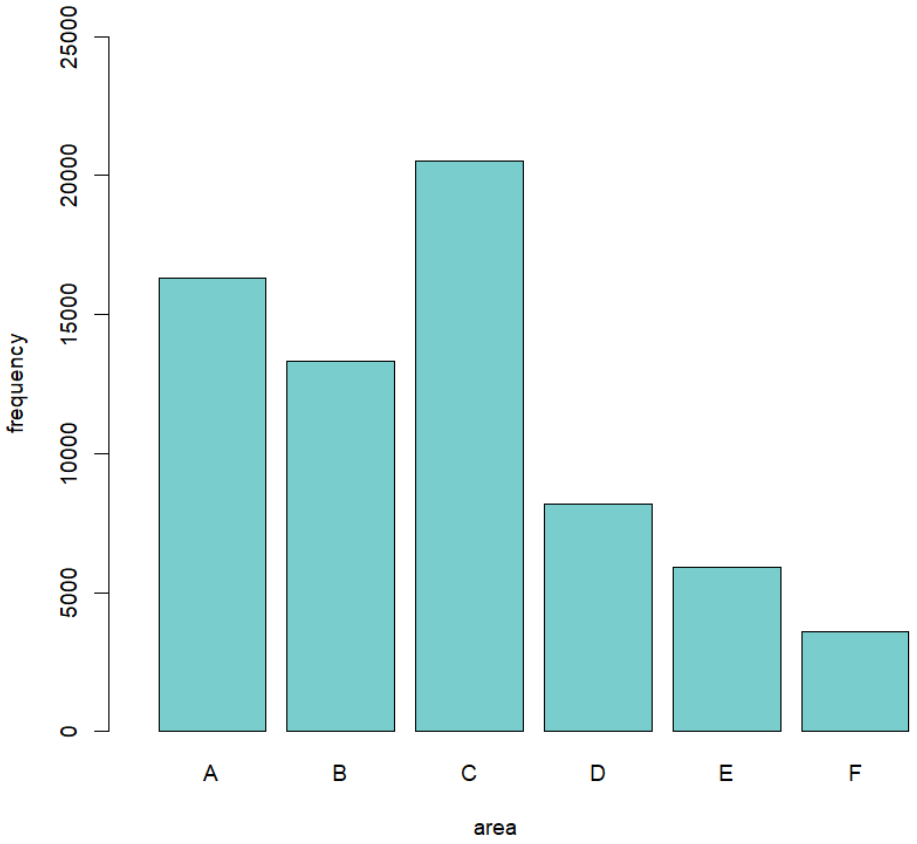
Boxplot of `veh_age`:



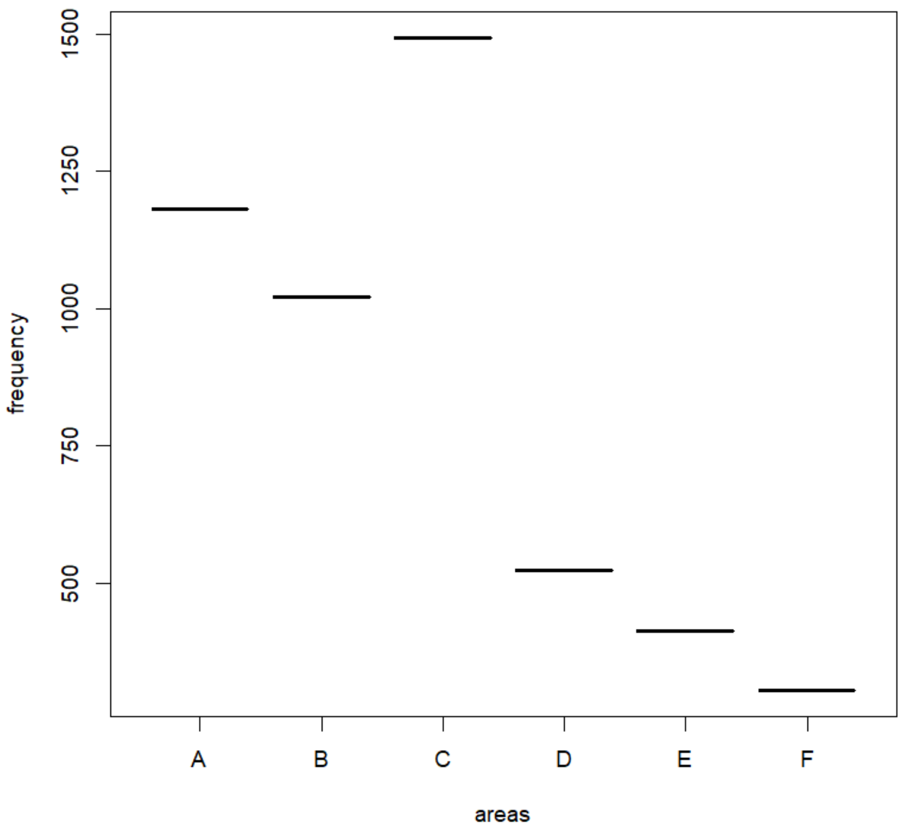
Distribution of `gender`:



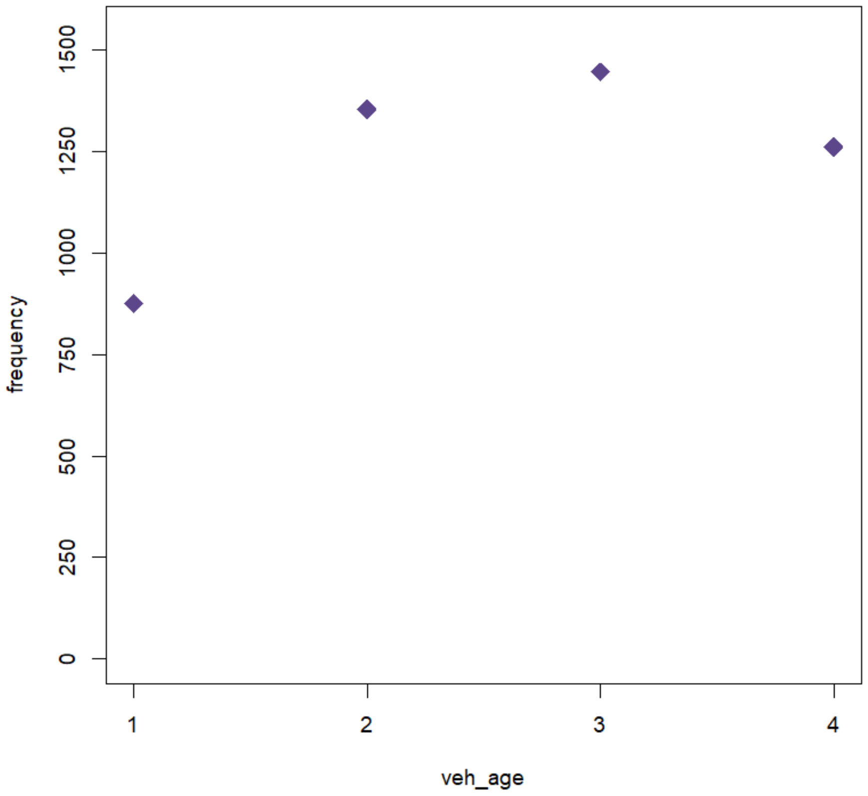
Distribution of `area`:



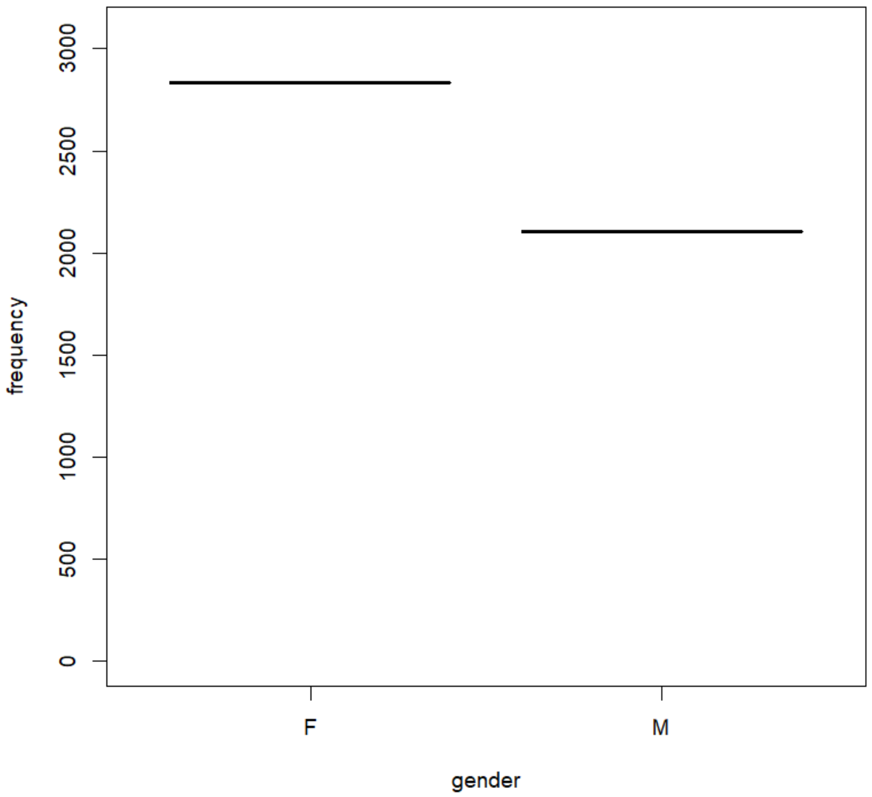
Nr. of claims per `area`:



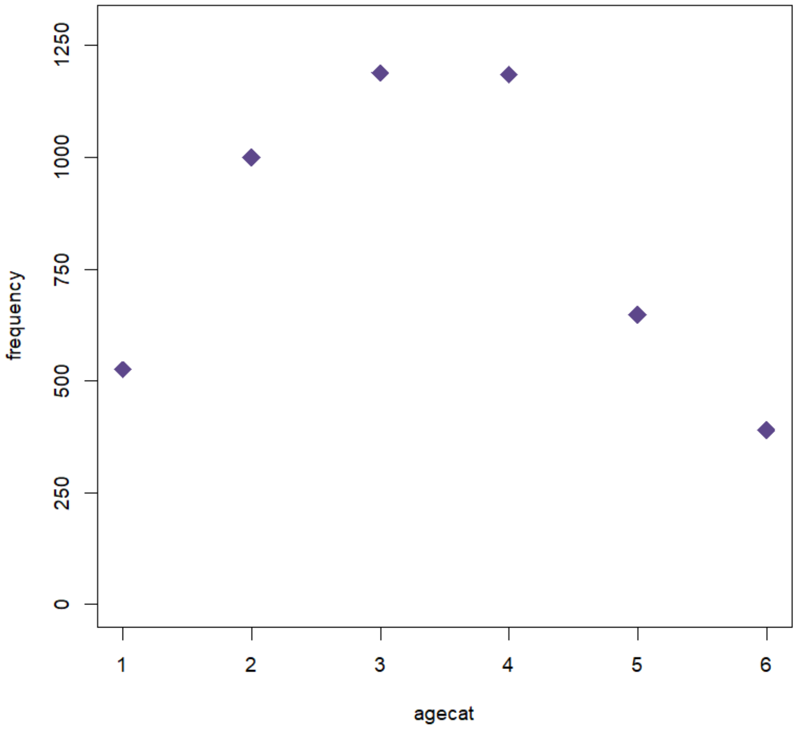
Nr. of claims per `veh_age`:



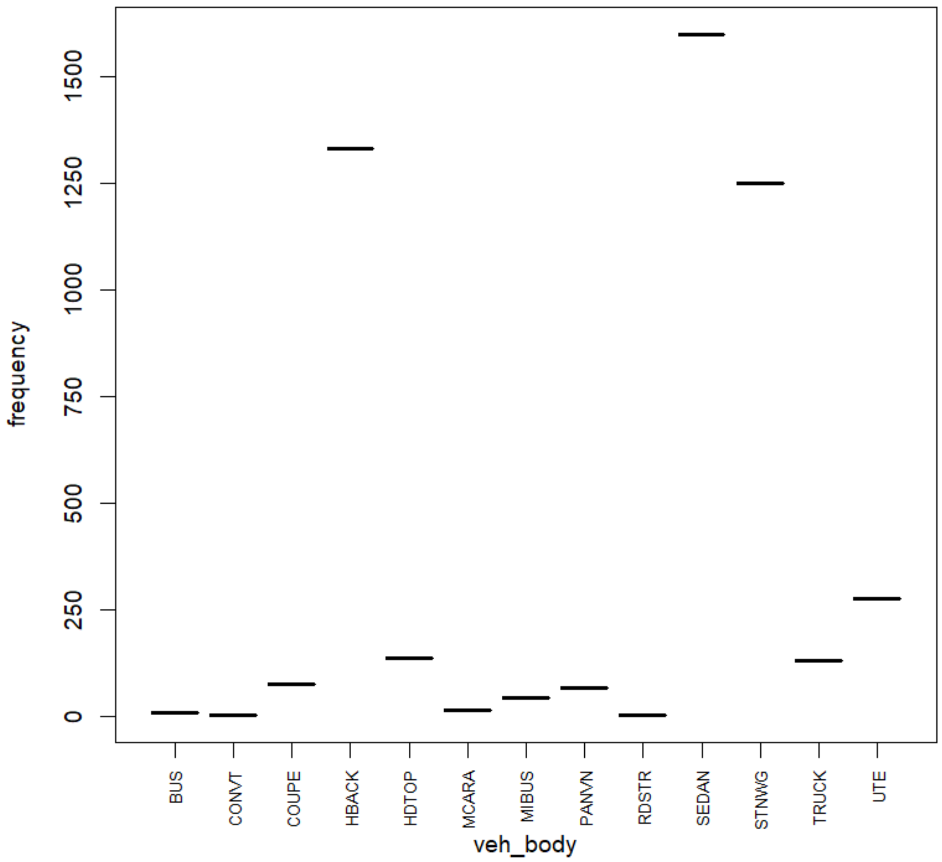
Nr. of claims per `gender`:



Nr. of claims per `age_cat`:



Nr. of claims per `veh_body`:



```
> total_claims
[1] 4937
> total_claims_amount
[1] 9314604
> amount_per_claim_overallData
[1] 137.2702
> amount_per_claim_claimsData
[1] 1886.693
```

Compare the observed mean to the observed variance of the number of claims.

Which one is larger? Test for overdispersion in R. One way to check for overdispersion is to run a quasi-poisson model, which fits an extra dispersion parameter to account for the extra variance. Then argue whether Poisson distribution or Negative Binomial distribution is to be used to fit our data.

Fit both models in R using glm (generalized linear model). Note that an adjustment for the exposure variable needs to be done. An offset in the glm function has to be used. This concept relates to the fact that if a one-year policy (exposure=1) has 2 claims, one would expect that a half-year policy (exposure=0.5) has 1 claim.[3]

If E[X]==Var[X], then using Poisson distribution is appropriate. If not, then other options need to be chosen, e.g. negative binomial distribution can be taken or an unrestricted dispersion parameter (Quasi-Poisson) model can be used.

In this case, the mean of `numclaims` is slightly smaller than the variance.

```
> meanOfNumclaims_observed
[1] 0.07275701
> varianceOfNumclaims_observed
[1] 0.07739737
```

Using a **Poisson regression model** yields:

```
> summary(model_poisson)

Call:
glm(formula = dataCar$numclaims ~ offset(log(dataCar$exposure)) + 
    dataCar$area + dataCar$gender, family = poisson)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-0.5981  -0.4574  -0.3497  -0.2243   4.3789  

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)     -1.844770   0.031583 -58.410   <2e-16 ***
dataCar$areaB    0.041809   0.042734   0.978   0.3279    
dataCar$areaC    0.002346   0.038943   0.060   0.9520    
dataCar$areaD   -0.125472   0.052491  -2.390   0.0168 *  
dataCar$areaE   -0.041666   0.057170  -0.729   0.4661    
dataCar$areaF    0.124319   0.064246   1.935   0.0530 .  
dataCar$genderM -0.038582   0.028791  -1.340   0.1802    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 25507  on 67855  degrees of freedom
Residual deviance: 25490  on 67849  degrees of freedom
AIC: 34938

Number of Fisher Scoring iterations: 6
```

[3] Sources refer to the whole chapter: https://www.tutorialspoint.com/r/r_poisson_regression.htm; https://www.statmethods.net/advstats/glm.html; https://biometry.github.io/APES/LectureNotes/2016-JAGS/Overdispersion/OverdispersionJAGS.pdf.

```
> # goodness-of-fit test (p>0.05)
> 1-pchisq(model_poisson$deviance,model_poisson$df.residual)
[1] 1
```

The goodness-of-fit test (chi-squared test) indicates that the Poisson regression model fits the data.

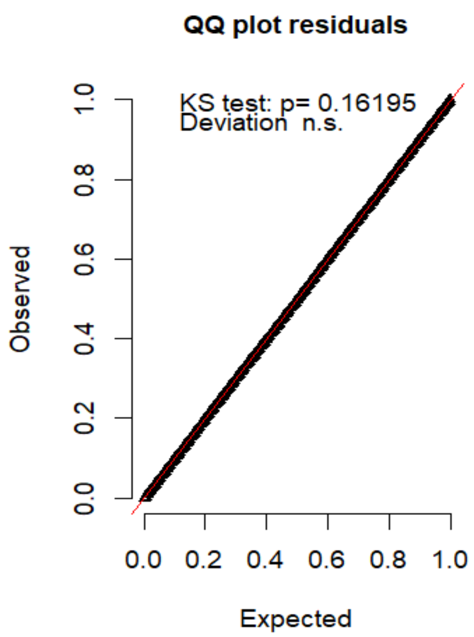
Using a **Quasi-Poisson regression model** yields:

```
> summary(model_quasipoisson)

Call:
glm(formula = dataCar$numclaims ~ offset(log(dataCar$exposure)) +
    dataCar$area + dataCar$gender, family = quasipoisson)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5981  -0.4574  -0.3497  -0.2243   4.3789

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -1.844770   0.037569 -49.104   <2e-16 ***
dataCar$areaB    0.041809   0.050833   0.822   0.4108
dataCar$areaC    0.002346   0.046324   0.051   0.9596
dataCar$areaD   -0.125472   0.062439  -2.010   0.0445 *
dataCar$areaE   -0.041666   0.068004  -0.613   0.5401
dataCar$areaF    0.124319   0.076422   1.627   0.1038
dataCar$genderM -0.038582   0.034248  -1.127   0.2599
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 1.41496)

    Null deviance: 25507  on 67855  degrees of freedom
Residual deviance: 25490  on 67849  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 6


> # ratio of residual deviance to redisual dfs should be ca. 1 (i.e. taken
dispersion param.)
> model_quasipoisson$deviance / model_quasipoisson$df.residual
[1] 0.375683


> # goodness-of-fit test (p>0.05)
> 1-pchisq(model_quasipoisson$deviance,model_quasipoisson$df.residual) # fi
ts here
[1] 1
```

The library AER can be used to perform a dispersion test:

```
> dispersiontest(model_poisson) # true dispersion is slightly >1 acc. to test

        Overdispersion test

data:  model_poisson
z = 4.4524, p-value = 4.246e-06
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.031659


> dispersiontest(model_poisson, trafo=1) # equidispersion: 0 (near 0 in this case)

        Overdispersion test

data:  model_poisson
z = 4.4524, p-value = 4.246e-06
alternative hypothesis: true alpha is greater than 0
sample estimates:
     alpha
0.03165909
```

The library DHARMa can also be used to analyze dispersion:




```
        DHARMa nonparametric dispersion test via mean deviance residual fitted vs.
        simulated-refitted

data:  simulationOutput
dispersion = 1, p-value = 1
alternative hypothesis: two.sided
```

Using a **Negative Binomial regression model** yields:

```
> summary(model_negativeBinomial) # lower residual deviance than poisson models

Call:
glm.nb(formula = dataCar$numclaims ~ offset(log(dataCar$exposure)) +
    dataCar$area + dataCar$gender, data = dataCar, init.theta = 2.06038555,
    link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5863  -0.4522  -0.3476  -0.2239   4.0137

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.84365    0.03232 -57.043   <2e-16 ***
dataCar$areaB    0.04331    0.04374   0.990   0.3221
dataCar$areaC    0.00385    0.03984   0.097   0.9230
dataCar$areaD   -0.12405    0.05359  -2.315   0.0206 *
dataCar$areaE   -0.03983    0.05842  -0.682   0.4954
dataCar$areaF    0.12547    0.06591   1.904   0.0569 .
dataCar$genderM -0.03864    0.02945  -1.312   0.1895
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(2.0604) family taken to be 1)

    Null deviance: 23420  on 67855  degrees of freedom
Residual deviance: 23404  on 67849  degrees of freedom
AIC: 34895

Number of Fisher Scoring iterations: 1


              Theta:  2.060
          Std. Err.:  0.357

 2 x log-likelihood:  -34879.159

> # goodness-of-fit test (p>0.05)
> 1-pchisq(model_negativeBinomial$deviance,model_negativeBinomial$df.residu
al) # fits here
[1] 1


> testOvrdispersion(sim_nb)

        DHARMa nonparametric dispersion test via mean deviance residual fit
ted vs.
        simulated-refitted

data:  simulationOutput
dispersion = 1, p-value < 2.2e-16
alternative hypothesis: two.sided
```

Understand and explain the concept of zero-inflated distribution and fit it to the data.[4]

https://rdrr.io/rforge/countreg/man/zeroinfl.html

Zero-inflated Poisson regression model:

```
> summary(model_zeroinfl_poisson)

Call:
zeroinfl(formula = dataCar$numclaims ~ offset(log(dataCar$exposure)) + dataCar$area + dataCar$gender,
    dist = "poisson")

Pearson residuals:
    Min      1Q  Median      3Q     Max
-0.3641 -0.3111 -0.2590 -0.1787 42.8830

Count model coefficients (poisson with log link):
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.526811   0.088102 -17.330   <2e-16 ***
dataCar$areaB    0.015143   0.106985   0.142    0.887
dataCar$areaC    0.007783   0.101139   0.077    0.939
dataCar$areaD   -0.088749   0.133854  -0.663    0.507
dataCar$areaE    0.135928   0.144254   0.942    0.346
dataCar$areaF    0.216894   0.170241   1.274    0.203
dataCar$genderM -0.021994   0.076385  -0.288    0.773

Zero-inflation model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.48827    0.30904  -1.580    0.114
dataCar$areaB   -0.11834    0.38397  -0.308    0.758
dataCar$areaC    0.01138    0.34985   0.033    0.974
dataCar$areaD    0.12294    0.43891   0.280    0.779
dataCar$areaE    0.54704    0.40144   1.363    0.173
dataCar$areaF    0.28387    0.51148   0.555    0.579
dataCar$genderM  0.04753    0.25326   0.188    0.851
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 43
Log-likelihood: -1.74e+04 on 14 Df
```

[4] Sources refer tot he whole chapter: https://pareonline.net/getvn.asp?v=21&n=2; https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3238139/.

Zero-inflated Negative Binomial regression model:

```
> summary(model_zeroinfl_nb) # log(theta) estimation is not significant

Call:
zeroinfl(formula = dataCar$numclaims ~ offset(log(dataCar$exposure)) + dataCar$area + dataCar$gender,
    dist = "negbin")

Pearson residuals:
    Min      1Q  Median      3Q     Max 
-0.3642 -0.3111 -0.2589 -0.1785 42.9297 

Count model coefficients (negbin with log link):
                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)     -1.529586   0.093508 -16.358   <2e-16 ***
dataCar$areaB    0.015340   0.106889   0.144    0.886    
dataCar$areaC    0.008104   0.101126   0.080    0.936    
dataCar$areaD   -0.088165   0.133844  -0.659    0.510    
dataCar$areaE    0.135860   0.144074   0.943    0.346    
dataCar$areaF    0.216566   0.170134   1.273    0.203    
dataCar$genderM -0.021887   0.076328  -0.287    0.774    
Log(theta)       5.052544  11.158018   0.453    0.651    

Zero-inflation model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.49882    0.33328  -1.497    0.134
dataCar$areaB   -0.11840    0.38605  -0.307    0.759
dataCar$areaC    0.01264    0.35210   0.036    0.971
dataCar$areaD    0.12571    0.44183   0.285    0.776
dataCar$areaE    0.54973    0.40452   1.359    0.174
dataCar$areaF    0.28468    0.51422   0.554    0.580
dataCar$genderM  0.04830    0.25466   0.190    0.850
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Theta = 156.4199 
Number of iterations in BFGS optimization: 52 
Log-likelihood: -1.74e+04 on 15 Df
```

Zero-inflated distribution in general:

The concept of zero-inflated distribution is used in cases where the occurrence of the count events is very rare among the involved observations. For this reason, only few variables that are non-zero occur among the observations. Zero-inflated (regression) models (e.g. for Poisson or Negative Binomial distribution) take the high number of zeros in the distribution into account by incorporating the fact that high numbers of zeros occur in the outcome variable (Y). For the fitting, such methods use a mixture model that brings together several distributions. In our case (of Poisson and Negative Binomial regression), two distributions are combined. Firstly, one of the models uses logistic regression to predict the non-occurrence of events, which means the zeros. Secondly, one of the models analyzes the frequency of occurrence of events (based on the condition that the event count is non-zero for the considered case). From this result two different sorts of coefficients. The zero-inflated model is a good option if the Poisson or Negative Binomial distribution model would predict too few zeros.

Compare all your candidate models using AIC. See also Vuong test.[5]

https://www.rdocumentation.org/packages/pscl/versions/1.5.2/topics/vuong

The AIC for all models:

```
> aic_models
         aic_poisson                aic_nb aic_zeroinfl_poisson      aic_zeroinfl_nb
            34938.42              34895.16             34831.88             34833.88
```

The AIC of the zero-inflated Poisson model is the smallest among all four models:

```
> min_AIC_model
aic_zeroinfl_poisson
            34831.88
```

Comparing Poisson vs. zero-inflated Poisson model by using Vuong test:

```
> vuong(model_poisson,model_zeroinfl_poisson)
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A    p-value
Raw                   -5.173272 model2 > model1 1.1501e-07
AIC-corrected         -4.572393 model2 > model1 2.4109e-06
BIC-corrected         -1.830835 model2 > model1   0.033563
```

Comparing Negative Binomial vs. zero-inflated Negative Binomial model by using Vuong test:

```
> vuong(model_negativeBinomial,model_zeroinfl_nb)
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A    p-value
Raw                  -4.0961017 model2 > model1 2.1008e-05
AIC-corrected        -3.3343604 model2 > model1 0.00042748
BIC-corrected         0.1411392 model1 > model2 0.44387999
```

The Vuong test (for non-nested models) has the H0 that the two models under consideration are equally close to the true distribution vs. the H1. However, it does not make a statement about whether the better of these two models is the truly "best" model.

[5] Sources refer to the whole chapter: https://www.theanalysisfactor.com/zero-inflated-poisson-models-for-count-outcomes/; http://cybermetrics.wlv.ac.uk/paperdata/misusevuong.pdf.

R Code:

```
######################################################################
# Hochdimensionale Statistik
######################################################################

# Vorbereitung:
install.packages("insuranceData")
install.packages("dplyr")
require(insuranceData)
require(dplyr)
rm(list=ls())
setwd("C:/Users/Coala/Desktop/HOCHDIM")

#----------------------------------------------------------------------

# 1.a. Get acquainted with dataset DataCar:
data("dataCar")
head(dataCar, n=10)

nrow(dataCar)
ncol(dataCar)

#----------------------------------------------------------------------

# 1.b. (see report)

#----------------------------------------------------------------------
# 1.c. Display descriptive statistics of interest about the number and
#      distributions of claims
#      (also across different areas, vehicle ages, etc.)

summary(dataCar) # min, 1st quantile, median, mean, 3rd quantile, max

boxplot(dataCar$veh_value)
sd(dataCar$veh_value)
range(dataCar$veh_value)

boxplot(dataCar$exposure)
sd(dataCar$exposure)
range(dataCar$exposure)

sum(dataCar$clm)/length(dataCar$clm) # % of claims based on entire dataset

boxplot(dataCar$claimcst0)
sd(dataCar$claimcst0)
range(dataCar$claimcst0)

table_veh <- table(dataCar$veh_body)
barplot(table_veh, cex.axis=1, xlab="vehicle body", ylab="frequency",
        col="darkslategray3", cex.names=0.7,
        xlim=c(0.5,length(table_veh)), ylim=c(0,25000))

boxplot(dataCar$veh_age)
sd(dataCar$veh_age)
range(dataCar$veh_age)
```

```
table_gender <- table(dataCar$gender)
barplot(table_gender, cex.axis=1, xlab="gender", ylab="frequency",
        col="darkslategray3", cex.names=1,
        xlim=c(0,length(table_gender)+0.7), ylim=c(0,40000))

table_area <- table(dataCar$area)
barplot(table_area, cex.axis=1, xlab="area", ylab="frequency",
        col="darkslategray3", cex.names=1,
        xlim=c(0,length(table_area)+0.7), ylim=c(0,25000))

table_agecat <- table(dataCar$agecat)
barplot(table_agecat, cex.axis=1, xlab="agecat", ylab="frequency",
        col="darkslategray3", cex.names=1,
        xlim=c(0,length(table_agecat)+0.7), ylim=c(0,20000))


# nr. of claims per area:
clm_by_area <- aggregate(dataCar$numclaims, by=list(dataCar$area), FUN=sum)
colnames(clm_by_area) <- c("area","clms")
clm_by_area["mean"] <- aggregate(dataCar$numclaims, by=list(dataCar$area), FUN=mean)[2]
clm_by_area["sd"] <- aggregate(dataCar$numclaims, by=list(dataCar$area), FUN=sd)[2]
plot(clm_by_area[1:2], ylab="frequency", xlab="areas", yaxt="n")
axis(2, at = seq(0, max(clm_by_area[2])+200, by = 250))

# nr. of claims per veh_age:
clm_by_vehicle_age <- aggregate(dataCar$numclaims, by=list(dataCar$veh_age), FUN=sum)
colnames(clm_by_vehicle_age) <- c("veh_age","clms")
clm_by_vehicle_age["mean"] <- aggregate(dataCar$numclaims, by=list(dataCar$veh_age), FUN=mean)[2]
clm_by_vehicle_age["sd"] <- aggregate(dataCar$numclaims, by=list(dataCar$veh_age), FUN=sd)[2]
plot(clm_by_vehicle_age[1:2], ylab="frequency", xlab="veh_age", xaxt="n", yaxt="n",
     pch=18, cex=2, col="mediumpurple4", ylim=c(0,max(clm_by_vehicle_age[2])+100))
axis(1, at = seq(1, nrow(clm_by_vehicle_age), by = 1))
axis(2, at = seq(0, max(clm_by_vehicle_age[2])+200, by = 250))

# nr. of claims per gender:
clm_by_gender <- aggregate(dataCar$numclaims, by=list(dataCar$gender), FUN=sum)
colnames(clm_by_gender) <- c("gender","clms")
plot(clm_by_gender[1:2], ylab="frequency", xlab="gender",
     ylim=c(0, max(clm_by_gender[2])+250))


# nr. of claims per agecat:
clm_by_agecat <- aggregate(dataCar$numclaims, by=list(dataCar$agecat), FUN=sum)
colnames(clm_by_agecat) <- c("agecat","clms")
clm_by_agecat["mean"] <- aggregate(dataCar$numclaims, by=list(dataCar$agecat), FUN=mean)[2]
clm_by_agecat["sd"] <- aggregate(dataCar$numclaims, by=list(dataCar$agecat), FUN=sd)[2]
plot(clm_by_agecat[1:2], ylab="frequency", xlab="agecat", xaxt="n", yaxt="n",
     pch=18, cex=2, col="mediumpurple4", ylim=c(0,max(clm_by_agecat[2])+100))
axis(1, at = seq(1, nrow(clm_by_agecat), by = 1))
axis(2, at = seq(0, max(clm_by_agecat[2])+200, by = 250))


# nr. of claims per veh_body:
clm_by_veh_body <- aggregate(dataCar$numclaims, by=list(dataCar$veh_body), FUN=sum)
colnames(clm_by_veh_body) <- c("veh_body","clms")
plot(clm_by_veh_body[1:2], ylab="frequency", xlab="veh_body", xaxt="n", yaxt="n")
axis(1, at = seq(1, nrow(clm_by_veh_body), by = 1),
     labels=as.character(clm_by_veh_body$veh_body), cex.axis=0.7, las=2)
axis(2, at = seq(0, max(clm_by_veh_body[2])+200, by = 250))
```

```
# distribution of claim amounts (claimcst0)
total_claims <- sum(dataCar$numclaims)
total_claims_amount <- sum(dataCar$claimcst0)

amount_per_claim_overallData <- total_claims_amount/nrow(dataCar)
  # or: mean(dataCar$claimcst0)
amount_per_claim_claimsData <- total_claims_amount/total_claims


#-----------------------------------------------------------------------

# 1.d. Compare the observed mean to the observed variance of the number of claims.
#      Which one is larger? Test for overdispersion in R. One way to check for overdispersion
#      is to run a quasi-poisson model, which fits an extra dispersion parameter to
#      account for the extra variance. Then argue whether Poisson distribution or Negative
#      Binomial distribution is to be used to fit our data.
# 1.e. Fit both models in R using glm (generalized linear model). Note that an adjustment
#      for the exposure variable needs to be done. An offset in the glm function has to be
#      used. This concept relates to the fact that if a one-year policy (exposure=1) has 2
#      claims, one would expect that a half-year policy (exposure=0.5) has 1 claim.

# mean vs. variance of numclaims
meanOfNumclaims_observed <- mean(dataCar$numclaims)
varianceOfNumclaims_observed <- (sd(dataCar$numclaims))^2

# if E[x]==Var[x], then using poisson distribution is ok
# if E[x]!=Var[x], then using poisson distribution is not ok -> take negative binomal distr.
#                   (glm.nb to fit the model)

varianceOfNumclaims_observed > meanOfNumclaims_observed # variance is larger in this case


# try poisson model
model_poisson=glm(dataCar$numclaims~offset(log(dataCar$exposure))+dataCar$area+dataCar$gender,
                  family=poisson)
                  # +offset(log(sum(dataCar$numclaims)))
                  # offset=log(sum(dataCar$numclaims))
summary(model_poisson)

# goodness-of-fit test (p>0.05)
1-pchisq(model_poisson$deviance,model_poisson$df.residual) # fits here


# test for overdispersion (using quasi-poisson model with extra dispersion parameter)
model_quasipoisson=glm(dataCar$numclaims~offset(log(dataCar$exposure))+dataCar$area+dataCar$gender,
                  family=quasipoisson)
summary(model_quasipoisson)

# ratio of residual deviance to redisual dfs should be ca. 1 (i.e. taken dispersion param.)
model_quasipoisson$deviance / model_quasipoisson$df.residual


print(1-pnorm(model_quasipoisson$deviance, model_quasipoisson$df.residual))
qqnorm(resid(model_quasipoisson))

# goodness-of-fit test (p>0.05)
1-pchisq(model_quasipoisson$deviance,model_quasipoisson$df.residual) # fits here
```

```
#install.packages("AER")
library(AER)
dispersiontest(model_poisson) # true dispersion is slightly >1 acc. to test
dispersiontest(model_poisson, trafo=1) # equidispersion: 0 (near 0 in this case)


#install.packages("devtools")
#install.packages("DHARMa")
library(DHARMa)
sim_model_poisson <- simulateResiduals(model_poisson, refit=T)
testOverdispersion(sim_model_poisson)
plotSimulatedResiduals(sim_model_poisson)


# try negative binomial distribution
#install.packages("MASS")
library(MASS)
model_negativeBinomial <- glm.nb(dataCar$numclaims~offset(log(dataCar$exposure))+
                                 dataCar$area+dataCar$gender,
                                 data=dataCar)
summary(model_negativeBinomial) # lower residual deviance than poisson models

# goodness-of-fit test (p>0.05)
1-pchisq(model_negativeBinomial$deviance,model_negativeBinomial$df.residual) # fits here

sim_nb <- simulateResiduals(model_negativeBinomial, refit=T,n=99)
plotSimulatedResiduals(sim_nb)

testOverdispersion(sim_nb)

#----------------------------------------------------------------------

# 1.f. Understand and explain the concept of zero-inflated distribution and fit it to the
#      data. https://rdrr.io/rforge/countreg/man/zeroinfl.html

#install.packages("pscl")
library(pscl)

# zero-inflated poisson model
model_zeroinfl_poisson=zeroinfl(dataCar$numclaims~offset(log(dataCar$exposure))+dataCar$area+
                                dataCar$gender, dist="poisson")
summary(model_zeroinfl_poisson)

# zero-inflated negative binomial model
model_zeroinfl_nb=zeroinfl(dataCar$numclaims~offset(log(dataCar$exposure))+dataCar$area+
                                dataCar$gender, dist="negbin")
summary(model_zeroinfl_nb) # log(theta) estimation is not significant
                           # >> zero-inflated poisson model fits better

#----------------------------------------------------------------------
```

```
#----------------------------------------------------------------------

# 1.g. Compare all your candidate models using AIC. See also Vuong test
#      (from pscl package).
#      https://www.rdocumentation.org/packages/pscl/versions/1.5.2/topics/vuong

# calculate AIC for all models
aic_poisson <- AIC(model_poisson)
aic_nb <- AIC(model_negativeBinomial)
aic_zeroinfl_poisson <- AIC(model_zeroinfl_poisson)
aic_zeroinfl_nb <- AIC(model_zeroinfl_nb)
aic_models <- c(aic_poisson=aic_poisson,aic_nb=aic_nb,
                aic_zeroinfl_poisson=aic_zeroinfl_poisson,
                aic_zeroinfl_nb=aic_zeroinfl_nb)

# min. AIC among models
min_AIC_model <- min(aic_models)
names(min_AIC_model) <- names(which(aic_models==min_AIC_model))
min_AIC_model

# vuong test (model vs. zero-inflated model)
vuong(model_poisson,model_zeroinfl_poisson)
vuong(model_negativeBinomial,model_zeroinfl_nb)

#----------------------------------------------------------------------
```