Introduction to Statistics
An Applied 3-Day Hands-On Workshop with R

Lecture 10: Regression

November 2019
Version: October 15, 2019

Michael Schomaker
UMIT University, Institute of Public Health, MDM and HTA

Model Formulas

Doing regression in R requires the use of "model formulas" which state the outcome, covariates, the relationship between covariates, amongst others.

```
> #  all these are model formulae
> f1 <- formula(time~temperature)
> f2 <- time~temperature
> f3 <- as.formula(time~temperature)
> f4 <- as.formula("time~temperature")
>
> # Linear Model
> lm(time~temperature)

Coefficients:
(Intercept)  temperature
     59.754       -0.406

> # the same
> lm(f1)
> lm(f2)
> lm(f3)
> lm(f4)
```

Linear Regression

Attach data, or use option `data`:

```
> detach(pizza)
> lm(time~temperature) # error
Fehler in eval(predvars, data, env) : Objekt 'temperature'
     nicht gefunden
> lm(time~temperature, data=pizza) # works

Call:
lm(formula = time ~ temperature, data = pizza)

Coefficients:
(Intercept)  temperature
     59.754       -0.406
```

Formulae with multiple variables

Use '+':

```
> # multiple covariates
> lm(time~temperature+free_wine+operator+bill)

Call:
lm(formula = time ~ temperature + free_wine + operator +
    bill)

Coefficients:
    (Intercept)      temperature         free_wine
       40.73511         -0.22184          9.74584

 operatorMelissa             bill
         0.07937          0.13186
```

Standard Regression Commands (I)

```
m1 <- lm(time~temperature+free_wine+bill)
summary(m1)       # main output
confint(m1)       # confidence intervals
coefficient(m1)   # extract coefficients
residuals(m1)     # extract residuals
plot(m1)          # plot diagnostics
termplot(m1)      # plot coefficients - makes sense?
predict(m1)       # predict Y (with measured X)
predict.lm(m1)    # the same
nd <- as.data.frame(matrix(c(50,1,50),ncol=3,nrow=1,
     dimnames=list(NULL,c("temperature","free_wine","bill"
     ))))
predict(m1, newdata=nd)   # predict Y (with new data)
```

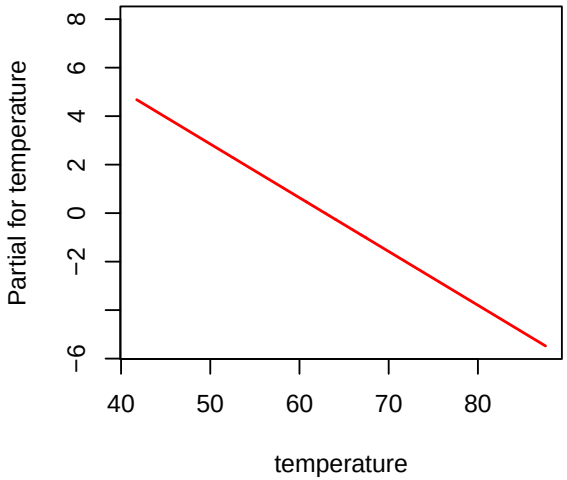
Standard Regression Commands (II)

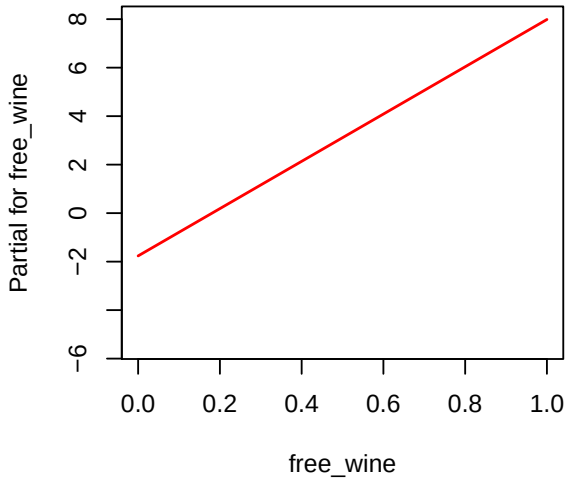

```
> summary(m1)      # main output

Call:
lm(formula = time ~ temperature + free_wine + bill)

Residuals:
     Min       1Q   Median       3Q      Max
-11.9689  -2.8128   0.0574   2.9353  11.8918

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 40.75604    1.43718   28.36   <2e-16 ***
temperature -0.22162    0.01852  -11.97   <2e-16 ***
free_wine    9.74867    0.30888   31.56   <2e-16 ***
bill         0.13196    0.01147   11.51   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' ' 1

Residual standard error: 4.117 on 1262 degrees of freedom
Multiple R-squared:  0.595,     Adjusted R-squared:
    0.5941
F-statistic: 618.1 on 3 and 1262 DF,  p-value: < 2.2e-16
```

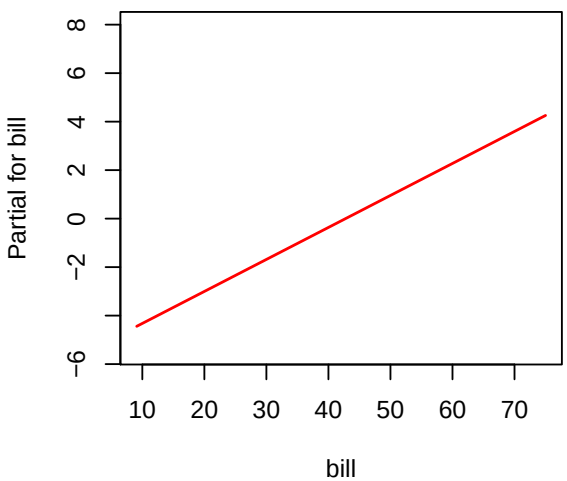
Standard Regression Commands (III)

```
> confint(m1)      # confidence intervals
                  2.5 %      97.5 %
(Intercept) 37.9365129 43.5755735
temperature -0.2579463 -0.1852926
free_wine    9.1426963 10.3546504
bill         0.1094582  0.1544584
```

Standard Regression Commands (IV) - `termplot`







Object Structure → use `str`

```
> # What is the structure of a regression object?
> str(m1)
List of 12
 $ coefficients : Named num [1:4] 40.756 -0.222 9.749
     0.132
  ..- attr(*, "names")= chr [1:4] "(Intercept)" "
     temperature" "free_wine" "bill"
 $ residuals    : Named num [1:1266] 1.7998 -3.3022
     -0.6949 -0.4454 0.0175 ...
  ..- attr(*, "names")= chr [1:1266] "1" "2" "3" "4" ...
 $ effects      : Named num [1:1266] -1.22e+03 9.97e+01
     -1.39e+02 4.74e+01 7.31e-02 ...
  ..- attr(*, "names")= chr [1:1266] "(Intercept)" "
     temperature" "free_wine" "bill" ...
 $ rank         : int 4
 $ fitted.values: Named num [1:1266] 33.3 28.5 46.3 29.8
     30 ...
  ..- attr(*, "names")= chr [1:1266] "1" "2" "3" "4" ...
 $ assign       : int [1:4] 0 1 2 3
 $ qr           :List of 5
  ..$ qr   : num [1:1266, 1:4] -35.5809 0.0281 0.0281
     0.0281 0.0281 ...
  .. ..- attr(*, "dimnames")=List of 2
 ...
```

Structure of the summary (I)

```
> m1$coefficients # instead of coefficients(m1)
> m1$residuals    # instead of residuals(m1)
>
> sm1 <- summary(m1)
> str(sm1)
List of 11
 $ call          : language lm(formula = time ~ temperature
      + free_wine + bill)
 $ terms         :Classes 'terms', 'formula'  language time
      ~ temperature + free_wine + bill
  .. ..- attr(*, "variables")= language list(time,
      temperature, free_wine, bill)
  .. ..- attr(*, "factors")= int [1:4, 1:3] 0 1 0 0 0 0 1
      0 0 0 ...
  .. .. ..- attr(*, "dimnames")=List of 2
...
```

Structure of the summary (II)


```
> sm1$coefficients # full table
               Estimate Std. Error   t value      Pr(>|t|)
(Intercept) 40.7560432 1.43718261  28.35829 2.827210e-137
temperature -0.2216195 0.01851666 -11.96865  2.400888e-31
free_wine    9.7486734 0.30888111  31.56125 1.212332e-161
bill         0.1319583 0.01146885  11.50580  3.331587e-29

> sm1$adj.r.squared
[1] 0.5940671
```

Create your own summary (I)


```
> # Make your own summary
> mysummary <- round(cbind(coefficients(m1),confint(m1),
    sm1$coefficients[,4]),digits=4)
> colnames(mysummary) <- c("Est.","LCI","UCI","pvalue")
> mysummary
                Est.     LCI     UCI pvalue
(Intercept) 40.7560 37.9365 43.5756      0
temperature -0.2216 -0.2579 -0.1853      0
free_wine    9.7487  9.1427 10.3547      0
bill         0.1320  0.1095  0.1545      0
```

Create your own summary (II)


```
> # Or even write your function:
> ms <- function(robject){
+ mysummary <- round(cbind(coefficients(robject),confint(
    robject),summary(robject)$coefficients[,4]),digits=4)
+ colnames(mysummary) <- c("Est.","LCI","UCI","pvalue")
+ return(mysummary)
+ }
> ms(m1)
                Est.     LCI     UCI pvalue
(Intercept)  40.7560 37.9365 43.5756      0
temperature -0.2216 -0.2579 -0.1853      0
free_wine    9.7487  9.1427 10.3547      0
bill         0.1320  0.1095  0.1545      0
```

Dummy coding with categorical variables

Just use a factor variable, that's all!

```
> ### Categorical variables
> # need to be a factor
> is.factor(free_wine) # not ideal
[1] FALSE
> is.factor(driver)    # Good
[1] TRUE
> lm(time~driver)

Call:
lm(formula = time ~ driver)

Coefficients:
    (Intercept)   driverDomenico      driverLuigi
        35.3128           -5.9964           -1.9338

   driverMario  driverSalvatore
       -0.8517           -0.6808

> lm(time~as.factor(driver)) # alternative
```

Change reference category with `relevel`


```
> lm(time~relevel(driver,ref="Luigi"))

Call:
lm(formula = time ~ relevel(driver, ref = "Luigi"))

Coefficients:

(Intercept)       relevel(driver, ref = "Luigi")Bruno
     33.379                                    1.934

relevel(driver, ref = "Luigi")Domenico
                                -4.063

relevel(driver, ref = "Luigi")Mario
                                1.082

relevel(driver, ref = "Luigi")Salvatore
                                  1.253
```

Categorical Variables – Manual Approach


```
> # Of course, can all be done manually too
> pizza$East <- as.numeric(branch=="East")
> pizza$West <- as.numeric(branch=="West")
> lm(time~East+West, data=pizza)

Call:
lm(formula = time ~ East + West, data = pizza)

Coefficients:
(Intercept)         East         West
     36.313       -5.246       -1.118

> lm(time~branch)                 # the same

Call:
lm(formula = time ~ branch)

Coefficients:
(Intercept)   branchEast   branchWest
     36.313       -5.246       -1.118
```

Output for categorical variable

```
> m3 <- lm(time~branch)
> summary(m3)

...

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  36.3127     0.2957 122.819  < 2e-16 ***
branchEast   -5.2461     0.4209 -12.463  < 2e-16 ***
branchWest   -1.1182     0.4148  -2.696  0.00711 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' '  1

Residual standard error: 6.066 on 1263 degrees of freedom
Multiple R-squared:  0.1199,    Adjusted R-squared:
    0.1185
F-statistic: 86.05 on 2 and 1263 DF,  p-value: < 2.2e-16
```

What do we want to test?

The hypothesis we may have in mind may be

$$H_0 : \mu_{\text{East}} = \mu_{\text{West}} = \mu_{\text{Centre}}$$

which corresponds to

$$H_0 : \beta_{\text{East}} = \beta_{\text{West}} = \beta_{\text{Centre}}$$

in the context of the regression model.

These are two identical hypotheses because in the regression setup, we are essentially comparing three conditional means $E(Y|X = x_1) = E(Y|X = x_2) = E(Y|X = x_3)$.

An ANOVA table summarizes the corresponding F-Test.

ANOVA table


```
> anova(m3) # test categorical variable
Analysis of Variance Table

Response: time
           Df Sum Sq Mean Sq F value    Pr(>F)
branch      2   6334  3166.8   86.05 < 2.2e-16 ***
Residuals 1263  46481    36.8
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' '  1

> anova(lm(time~branch+bill+driver))
Analysis of Variance Table

Response: time
           Df Sum Sq Mean Sq  F value    Pr(>F)
branch      2   6334  3166.8 100.2596 < 2.2e-16 ***
bill        1   6170  6170.4 195.3505 < 2.2e-16 ***
driver      4    575   143.7   4.5505  0.001187 **
Residuals 1258  39736    31.6
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' '  1
```

Transformations

```
> ### Transformations
> # use I()
> lm(temperature~time+time^2)     # time^2 omitted

Call:
lm(formula = temperature ~ time + time^2)

Coefficients:
(Intercept)         time
    78.7385      -0.4638

> lm(temperature~time+I(time^2)) # correct

Call:
lm(formula = temperature ~ time + I(time^2))

Coefficients:
(Intercept)         time    I(time^2)
  108.93354     -2.32176       0.02753
```

Interactions

Use '$*$' or '$:$' as part of the model formula:

```
> ### Interactions
> lm(time~temperature+bill+temperature:bill)

Coefficients:
     (Intercept)       temperature
       62.083345         -0.559729

            bill  temperature:bill
     -0.264921          0.007026
> lm(time~temperature*bill) #the same

Coefficients:
     (Intercept)       temperature
       62.083345         -0.559729

            bill  temperature:bill
     -0.264921          0.007026
```

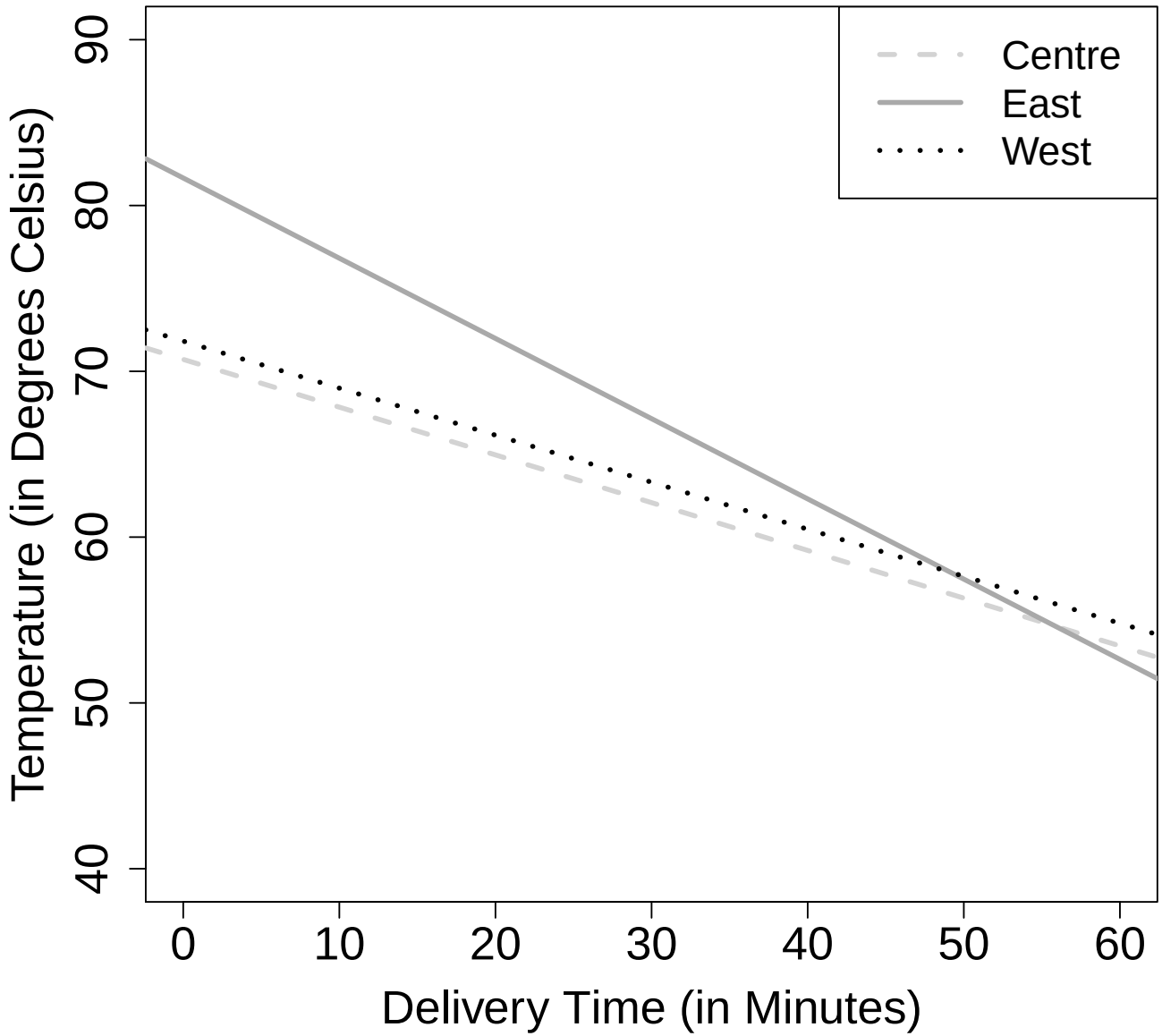
Categorical-continuous interaction


```
> # Categorical-Continuous Interaction
> int.m1 <- lm(temperature~time*branch)
> summary(int.m1)

Call:
lm(formula = temperature ~ time * branch)

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     70.718327   1.850918  38.207  < 2e-16 ***
time            -0.288011   0.050342  -5.721 1.32e-08 ***
branchEast      10.941411   2.320682   4.715 2.69e-06 ***
branchWest       1.102597   2.566087   0.430  0.66750
time:branchEast -0.195885   0.066897  -2.928  0.00347 **
time:branchWest  0.004352   0.070844   0.061  0.95103
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' '  1

Residual standard error: 5.951 on 1260 degrees of freedom
Multiple R-squared:  0.2602,    Adjusted R-squared:
    0.2573
F-statistic: 88.64 on 5 and 1260 DF,  p-value: < 2.2e-16
```

Categorical-continuous interaction (visualization)


```r
# Advanced: visualize interaction
c7 <- coefficients(int.m1)
par(mar= c(5, 5, 2, 2))
plot(-c(-5,5),cex=1.75,pch=19,xlim=c(0,60),ylim=c(40,90),
    ylab="Temperature (in Degrees Celsius)",xlab="
    Delivery Time (in Minutes)", cex.axis=1.75,cex.lab
    =1.75)
abline(a=c7[1],b=c7[2],lwd=3,col="lightgrey",lty=2)
abline(a=c7[1]+c7[3],b=c7[2]+c7[5],lwd=3,col="darkgrey",
    lty=1)
abline(a=c7[1]+c7[4],b=c7[2]+c7[6],lwd=3,col="black",lty
    =3)
legend("topright",col=c("lightgrey","darkgrey","black"),
    legend=c("Centre","East","West"),lwd=3,cex=1.5,lty=c
    (2,1,3))
```

Categorical-continuous interaction (visualization) II



Categorical-categorical interactions


```
> # categorical-categorical interactions
> summary(lm(time~branch*operator))

Coefficients:
                          Estimate ... Pr(>|t|)
(Intercept)                36.4203 ...   <2e-16 ***
branchEast                 -5.6685 ...   <2e-16 ***
branchWest                 -1.3599 ...   0.0205 *
operatorMelissa            -0.2178 ...   0.7129
branchEast:operatorMelissa  0.8599 ...   0.3076
branchWest:operatorMelissa  0.4842 ...   0.5598
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' ' 1

Residual standard error: 6.07 on 1260 degrees of freedom
Multiple R-squared:  0.121,     Adjusted R-squared:
    0.1175
F-statistic: 34.68 on 5 and 1260 DF,  p-value: < 2.2e-16
```

Continuous-continuous interactions

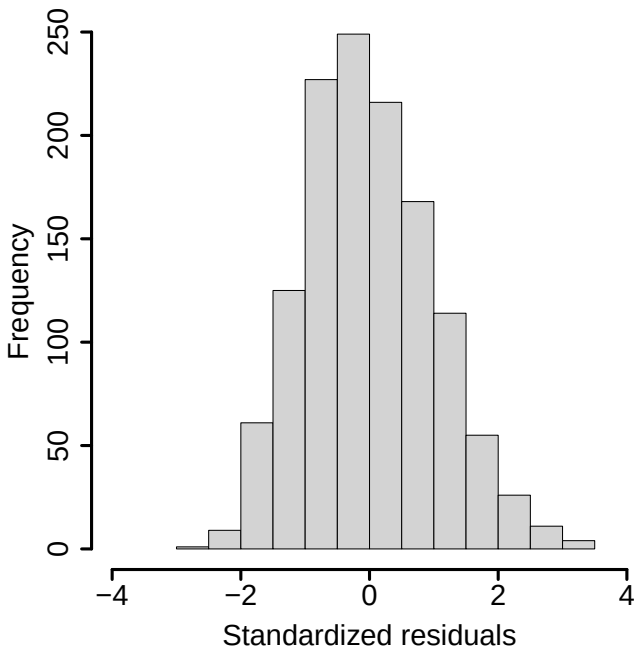

```
> # continuous-continuous interactions
> summary(lm(temperature~bill*time))

Coefficients:
             Estimate ... Pr(>|t|)
(Intercept) 92.555943 ...  < 2e-16 ***
bill        -0.454381 ... 4.34e-11 ***
time        -0.679537 ... 6.31e-15 ***
bill:time    0.008687 ... 1.89e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
    ' '  1

Residual standard error: 5.948 on 1262 degrees of freedom
Multiple R-squared:   0.26,     Adjusted R-squared:
    0.2582
F-statistic: 147.8 on 3 and 1262 DF,  p-value: < 2.2e-16
```

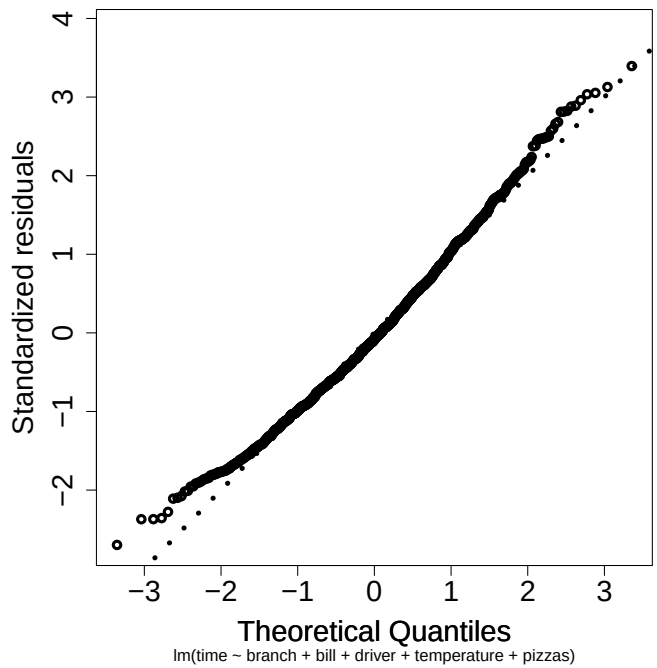
Basic Diagnostics (I)


```
> # basic checks for linear model
> plot(m1)
Warte auf Bestätigung des Seitenwechsels...
Warte auf Bestätigung des Seitenwechsels...
Warte auf Bestätigung des Seitenwechsels...
Warte auf Bestätigung des Seitenwechsels...
> plot(m1, which=2)     # QQ Plot
> plot(m1, which=3)     # check heteroskedasticity
> hist(residuals(m1))   # histogram of residuals
```

Checking the normality assumption

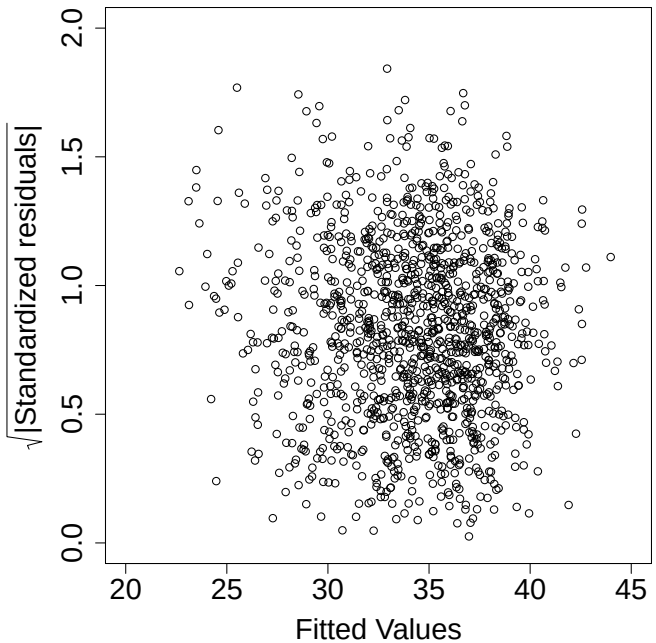
....using a histogram of the residuals and a QQ-Plot:

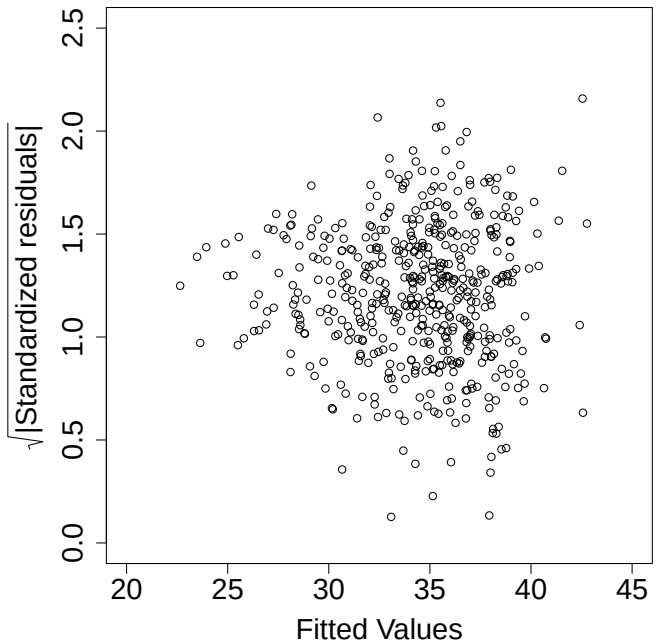




Checking heteroskedasticity

Left: good; Right: bad





Logistic Regression – `glm` and `family="binomial"`


```
> # logistic model
> m4 <- glm(free_wine~bill+driver+branch+operator, family=
    "binomial")
> summary(m4)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -2.767954   0.423956  -6.529 6.63e-11 ***
bill             0.038109   0.007785   4.895 9.81e-07 ***
driverDomenico  -0.784671   0.456808  -1.718   0.0858 .
driverLuigi     -0.560532   0.297897  -1.882   0.0599 .
driverMario     -0.260552   0.198818  -1.311   0.1900
driverSalvatore -0.128486   0.193854  -0.663   0.5075
branchEast      -0.922879   0.223016  -4.138 3.50e-05 ***
branchWest      -0.250446   0.167768  -1.493   0.1355
operatorMelissa  0.133901   0.152685   0.877   0.3805
---
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1197.0  on 1265  degrees of freedom
Residual deviance: 1110.9  on 1257  degrees of freedom
AIC: 1128.9
```

Logistic Regression – Odds Ratio scale


```
> # we need to exponentiate for odds ratio
> # a bit tiring....
> exp(coefficients(m4)) # odds ratio
(Intercept)             bill  driverDomenico
0.06279032      1.03884477      0.45626962

driverLuigi     driverMario driverSalvatore
 0.57090543      0.77062641      0.87942630

branchEast      branchWest operatorMelissa
0.39737349      0.77845374      1.14327965

> exp(confint(m4))      # CI for odds ratio
Waiting for profiling to be done...
                    2.5 %    97.5 %
(Intercept)     0.02707064 0.1428350
bill            1.02322261 1.0549563
driverDomenico  0.16859386 1.0415925
driverLuigi     0.31015996 1.0033954
driverMario     0.52093635 1.1369852
driverSalvatore 0.60087101 1.2859950
branchEast      0.25391033 0.6099795
branchWest      0.55959821 1.0808840
operatorMelissa 0.84773115 1.5432353
```

Poisson regression with `family="poisson"`

```
# Poisson model for count data or risk ratio scale
m5 <- glm(free_wine~bill+driver+branch+operator, family="
    poisson")
m6 <- glm(pizzas~bill+driver+branch+operator, family="
    poisson")
summary(m5)
summary(m6)
```

Additive models with penalized splines (I)

```
# additive models for penalized splines
library(mgcv)
m7 <- gam(time~s(temperature)+s(bill)+branch*operator)
summary(m7)
plot(m7)   #plot splines
```

Additive models with penalized splines (II)