Introduction to R

Regression in R

Jan-Philipp Kolb

10 März, 2020

Variables of the `mtcars` dataset

Help for the `mtcars` dataset:

```
?mtcars
```

- mpg - Miles/(US) gallon
- cyl - Number of cylinders
- disp - Displacement (cu.in.)
- hp - Gross horsepower
- drat - Rear axle ratio
- wt - Weight (1000 lbs)
- qsec - 1/4 mile time
- vs - Engine (0 = V-shaped, 1 = straight)
- am - Transmission (0 = automatic, 1 = manual)
- gear - Number of forward gears
- carb - Number of carburetors

Dataset `mtcars`

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3

Distributions of two variables of `mtcars`

```
par(mfrow=c(1,2))
plot(density(mtcars$wt)); plot(density(mtcars$mpg))
```

A simple regression model

Dependent variable - miles per gallon (mpg)

Independent variable - weight (wt)

```
m1 <- lm(mpg ~ wt,data=mtcars)
m1
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Coefficients:
## (Intercept)           wt
##      37.285       -5.344
```

Get the model summary

```
summary(m1)
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528,    Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

The model formula

Model without intercept

```
m2 <- lm(mpg ~ - 1 + wt,data=mtcars)
summary(m2)$coefficients
```

```
##    Estimate Std. Error  t value    Pr(>|t|)
## wt 5.291624  0.5931801 8.920771 4.55314e-10
```

Adding further variables

```
m3 <- lm(mpg ~ wt + cyl,data=mtcars)
summary(m3)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 39.686261  1.7149840 23.140893 3.043182e-20
## wt          -3.190972  0.7569065 -4.215808 2.220200e-04
## cyl         -1.507795  0.4146883 -3.635972 1.064282e-03
```

Estimation based on a subsample

```r
lm(mpg~wt+disp, data=mtcars, subset=(wt>3))
```

```
##
## Call:
## lm(formula = mpg ~ wt + disp, data = mtcars, subset = (wt > 3))
##
## Coefficients:
## (Intercept)           wt         disp
##    28.40497     -1.46360     -0.02016
```

- where only cars heavier than 3000 lb are considered.

The command `as.formula`

Creating a formula object

```
?as.formula
```

```
fo <- mpg ~ wt + cyl
```

```
class(fo)
```

```
## [1] "formula"
```

```
# The formula object can be used in the regression:
m3 <- lm(fo,data=mtcars)
```

Further possibilities to specify the formula

Take all available predictors

```
m3_a<-lm(mpg~.,data=mtcars)
```

Interaction effect

```
# effect of cyl and interaction effect:
m3a<-lm(mpg~wt*cyl,data=mtcars)

# only interaction effect:
m3b<-lm(mpg~wt:cyl,data=mtcars)
```

Take the logarithm

```
m3d<-lm(mpg~log(wt),data=mtcars)
```

Further transformations

Further transformations:

Tranformations of variables are directly included with the $I()$ function:

```
fo2 <- I(log(mpg))~wt+I(wt^2)+disp
lm(fo2, data=mtcars)
```

```
##
## Call:
## lm(formula = fo2, data = mtcars)
##
## Coefficients:
## (Intercept)           wt      I(wt^2)         disp
##   4.0000825   -0.3499056    0.0275548   -0.0009865
```

The command `setdiff`

- We can use the command to create a dataset with only the features, without the dependent variable

```
names(mtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"
##  [9] "am"   "gear" "carb"
```

```
features <- setdiff(names(mtcars), "mpg")
features
```

```
##  [1] "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"
##  [9] "gear" "carb"
```

```
featdat <- mtcars[,features]
```

The command `model.matrix`

- With `model.matrix` the qualitative variables are automatically dummy encoded

```
?model.matrix
```

```
model.matrix(m3d)
```

```
##                     (Intercept)   log(wt)
## Mazda RX4                     1 0.9631743
## Mazda RX4 Wag                 1 1.0560527
## Datsun 710                    1 0.8415672
## Hornet 4 Drive                1 1.1678274
## Hornet Sportabout             1 1.2354715
## Valiant                       1 1.2412686
## Duster 360                    1 1.2725656
## Merc 240D                     1 1.1600209
## Merc 230                      1 1.1474025
## Merc 280                      1 1.2354715
## Merc 280C                     1 1.2354715
## Merc 450SE                    1 1.4036430
## Merc 450SL                    1 1.3164082
## Merc 450SLC                   1 1.3297240
```

Model matrix (II)

- We can also create a model matrix directly from the formula and data arguments
- See `Matrix::sparse.model.matrix` for increased efficiency on large dimension data.

```
ff <- mpg ~ log(wt):cyl
m <- model.frame(ff, mtcars)
```

```
(mat <- model.matrix(ff, m))
```

```
##                     (Intercept) log(wt):cyl
## Mazda RX4                     1    5.779046
## Mazda RX4 Wag                 1    6.336316
## Datsun 710                    1    3.366269
## Hornet 4 Drive                1    7.006964
## Hornet Sportabout             1    9.883772
## Valiant                       1    7.447612
## Duster 360                    1   10.180525
## Merc 240D                     1    4.640084
## Merc 230                      1    4.589610
## Merc 280                      1    7.412829
## Merc 280C                     1    7.412829
```

A model with interaction effect

```
# disp    -  Displacement (cu.in.)
m3d<-lm(mpg~wt*disp,data=mtcars)
m3dsum <- summary(m3d)
m3dsum$coefficients
```

```
##                Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept) 44.08199770 3.123062627 14.114990 2.955567e-14
## wt          -6.49567966 1.313382622 -4.945763 3.216705e-05
## disp        -0.05635816 0.013238696 -4.257078 2.101721e-04
## wt:disp      0.01170542 0.003255102  3.596022 1.226988e-03
```

Residual plot - model assumptions violated?

- We have model assumptions violated if points deviate with a pattern from the line

```
plot(m3,1)
```

Residual plot

```
plot(m3,2)
```

Another example for object orientation

- `m3` is now a special regression object
- Various functions can be applied to this object

```r
predict(m3) # Prediction
resid(m3) # Residuals
```

```
##          Mazda RX4     Mazda RX4 Wag        Datsun 710
##           22.27914          21.46545          26.25203
##     Hornet 4 Drive Hornet Sportabout           Valiant
##           20.38052          16.64696          19.59873

##          Mazda RX4     Mazda RX4 Wag        Datsun 710
##         -1.2791447        -0.4654468        -3.4520262
##     Hornet 4 Drive Hornet Sportabout           Valiant
##          1.0194838         2.0530424        -1.4987281
```

Make model prediction

```
pre <- predict(m1)
head(mtcars$mpg)
```

```
## [1] 21.0 21.0 22.8 21.4 18.7 18.1
```

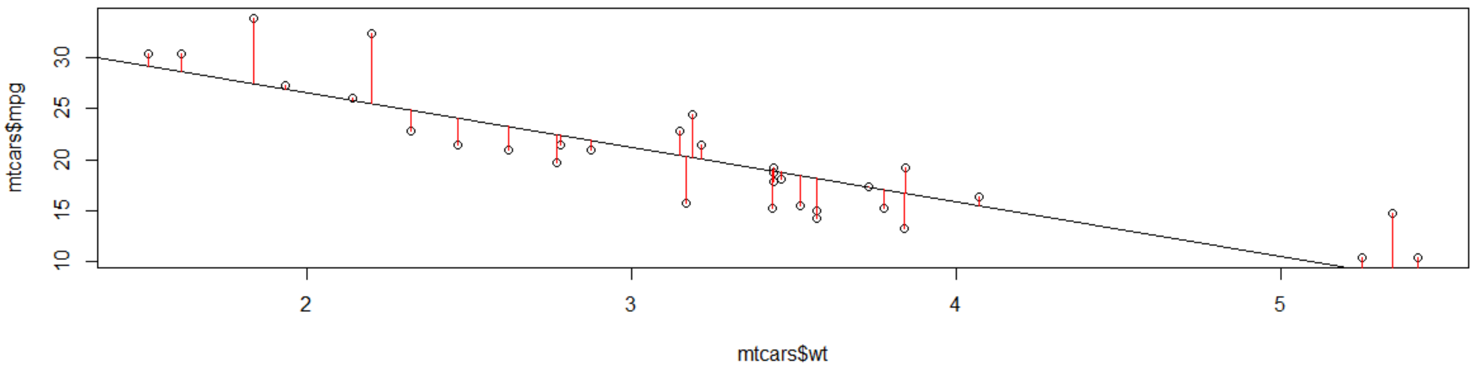
```
head(pre)
```

```
##         Mazda RX4     Mazda RX4 Wag        Datsun 710
##          23.28261          21.91977          24.88595
##    Hornet 4 Drive Hornet Sportabout           Valiant
##          20.10265          18.90014          18.79325
```

Regression diagnostic with base-R

Visualizing residuals

```
plot(mtcars$wt,mtcars$mpg)
abline(m1)
segments(mtcars$wt, mtcars$mpg, mtcars$wt, pre, col="red")
```



The bias-variance tradeoff (I)

The bias–variance tradeoff is the property of a set of predictive models whereby models with a lower bias in parameter estimation have a higher variance of the parameter estimates across samples, and vice versa.

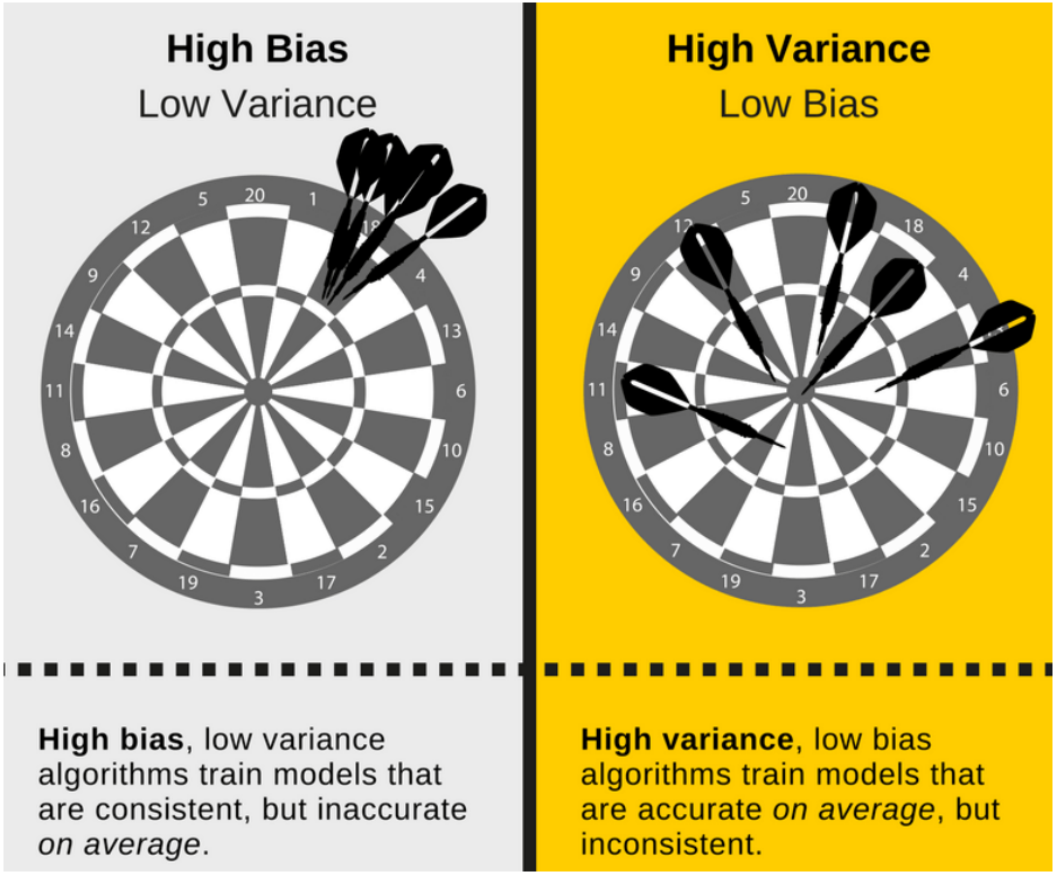
The bias error

… is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

The variance

… is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

The bias-variance tradeoff (II)



The mean squared error (mse)

- The **MSE** measures the average of the squares of the errors
- **The lower the better**

```
(mse5 <- mean((mtcars$mpg -  pre)^2)) # model 5
```

```
## [1] 8.697561
```

```
(mse3 <- mean((mtcars$mpg -  predict(m3))^2))
```

```
## [1] 5.974124
```

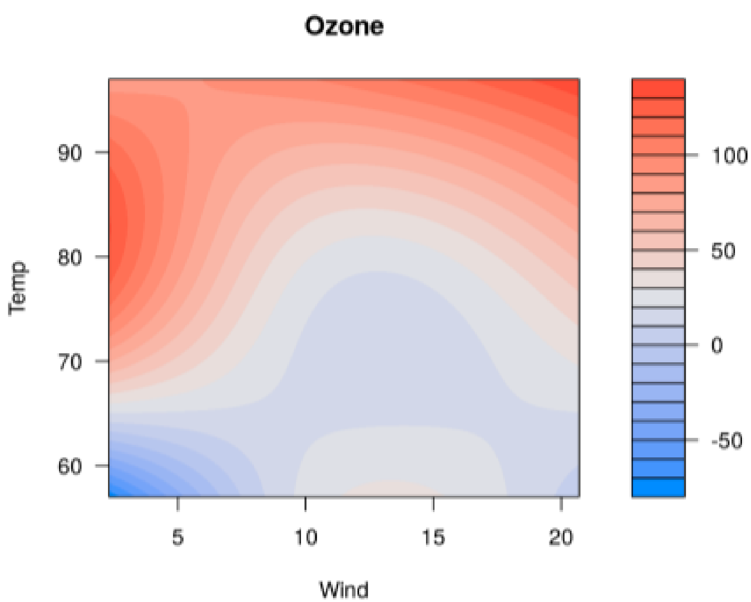
Package Metrics to compute mse

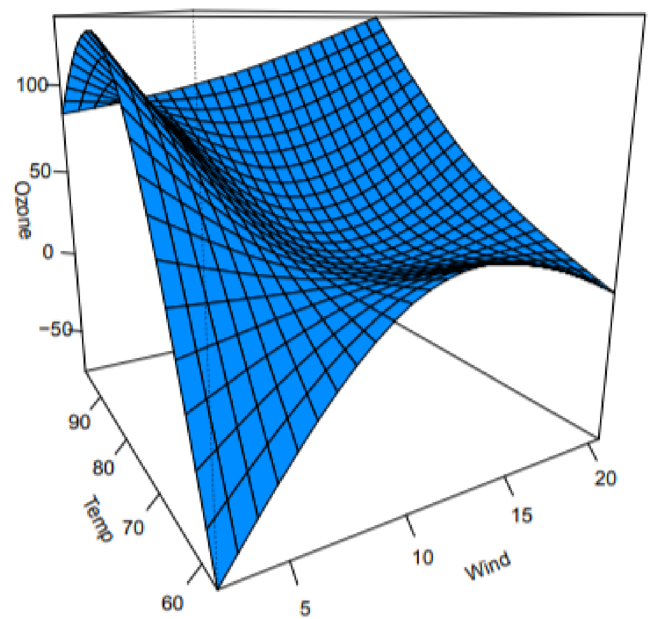
```
library(Metrics)
mse(mtcars$mpg,predict(m3))
```

```
## [1] 5.974124
```

The `visreg`-package

```r
library(visreg)
```

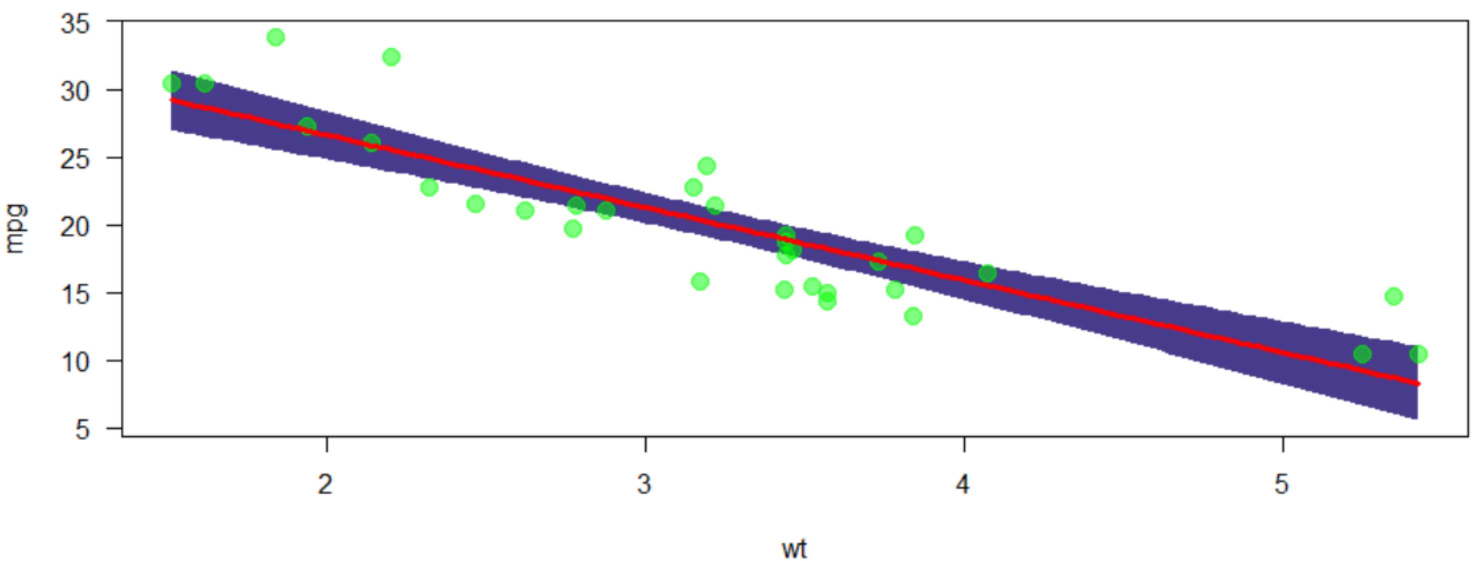




The `visreg`-package

- The default-argument for `type` is `conditional`.
- Scatterplot of `mpg` and `wt` plus regression line and confidence bands

```
visreg(m1, "wt", type = "conditional")
```



Regression with factors

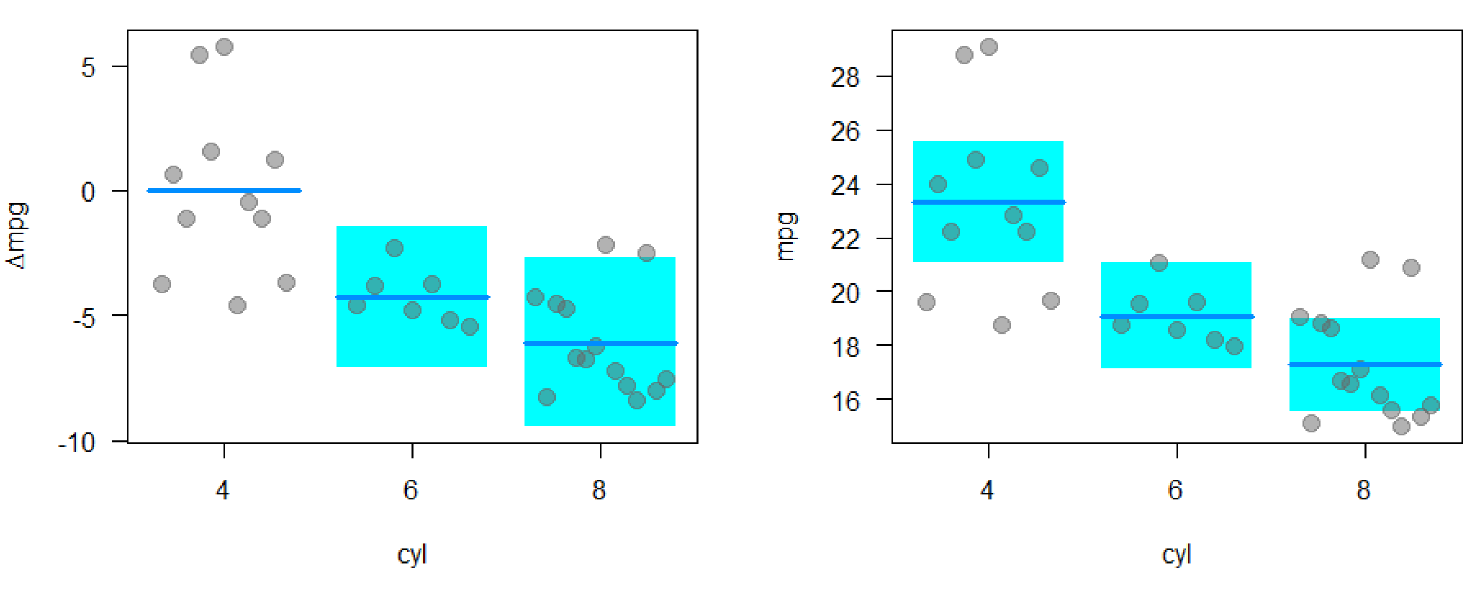
- The effects of factors can also be visualized with `visreg`:

```
mtcars$cyl <- as.factor(mtcars$cyl)
m4 <- lm(mpg ~ cyl + wt, data = mtcars)
# summary(m4)
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 33.990794  1.8877934 18.005569 6.257246e-17
## cyl6        -4.255582  1.3860728 -3.070244 4.717834e-03
## cyl8        -6.070860  1.6522878 -3.674214 9.991893e-04
## wt          -3.205613  0.7538957 -4.252065 2.130435e-04
```

Effects of factors

```
par(mfrow=c(1,2))
visreg(m4, "cyl", type = "contrast")
visreg(m4, "cyl", type = "conditional")
```



The package `visreg` - Interactions

```r
m5 <- lm(mpg ~ cyl*wt, data = mtcars)
# summary(m5)
```

```
##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  39.571196   3.193940 12.3894599 2.058359e-12
## cyl6        -11.162351   9.355346 -1.1931522 2.435843e-01
## cyl8        -15.703167   4.839464 -3.2448150 3.223216e-03
## wt           -5.647025   1.359498 -4.1537586 3.127578e-04
## cyl6:wt       2.866919   3.117330  0.9196716 3.661987e-01
## cyl8:wt       3.454587   1.627261  2.1229458 4.344037e-02
```

Control of the graphic output with `layout`.

```
visreg(m5, "wt", by = "cyl",layout=c(3,1))
```

The package `visreg` - Interactions overlay

```
m6 <- lm(mpg ~ hp + wt * cyl, data = mtcars)
```

```
visreg(m6, "wt", by="cyl", overlay=TRUE, partial=FALSE)
```

The package `visreg` - `visreg2d`

```
visreg2d(m6, "wt", "hp", plot.type = "image")
```

Exercise: regression Ames housing data

1) Install the package `AmesHousing` and create a **processed version** of the Ames housing data with (at least) the variables `Sale_Price`, `Gr_Liv_Area` and `TotRms_AbvGrd`

2) Create a regression model with `Sale_Price` as dependent and `Gr_Liv_Area` and `TotRms_AbvGrd` as independent variables. Then create seperated models for the two independent variables. Compare the results. What do you think?

The Ames Iowa Housing Data

```
ames_data <- AmesHousing::make_ames()
```

Some Variables

- `Gr_Liv_Area`: Above grade (ground) living area square feet
- `TotRms_AbvGrd`: Total rooms above grade (does not include bathrooms
- `MS_SubClass`: Identifies the type of dwelling involved in the sale.
- `MS_Zoning`: Identifies the general zoning classification of the sale.
- `Lot_Frontage`: Linear feet of street connected to property
- `Lot_Area`: Lot size in square feet
- `Street`: Type of road access to property
- `Alley`: Type of alley access to property
- `Lot_Shape`: General shape of property
- `Land_Contour`: Flatness of the propert

Multicollinearity

- As p increases we are more likely to capture multiple features that have some multicollinearity.
- When multicollinearity exists, we often see high variability in our coefficient terms.
- E.g. we have a correlation of 0.801 between `Gr_Liv_Area` and `TotRms_AbvGrd`
- Both variables are strongly correlated to the response variable (`Sale_Price`).

```
ames_data <- AmesHousing::make_ames()
cor(ames_data[,c("Sale_Price","Gr_Liv_Area","TotRms_AbvGrd")])
```

```
##               Sale_Price Gr_Liv_Area TotRms_AbvGrd
## Sale_Price     1.0000000   0.7067799     0.4954744
## Gr_Liv_Area    0.7067799   1.0000000     0.8077721
## TotRms_AbvGrd  0.4954744   0.8077721     1.0000000
```

Multicollinearity

```
lm(Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data = ames_data)
```

```
##
## Call:
## lm(formula = Sale_Price ~ Gr_Liv_Area + TotRms_AbvGrd, data = ames_data)
##
## Coefficients:
##   (Intercept)    Gr_Liv_Area  TotRms_AbvGrd
##       42767.6          139.4       -11025.9
```

- When we fit a model with both these variables we get a positive coefficient for `Gr_Liv_Area` but a negative coefficient for `TotRms_AbvGrd`, suggesting one has a positive impact to Sale_Price and the other a negative impact.

Seperated models

- If we refit the model with each variable independently, they both show a positive impact.
- The `Gr_Liv_Area` effect is now smaller and the `TotRms_AbvGrd` is positive with a much larger magnitude.

```
lm(Sale_Price ~ Gr_Liv_Area, data = ames_data)$coefficients
```

```
## (Intercept) Gr_Liv_Area
##   13289.634     111.694
```

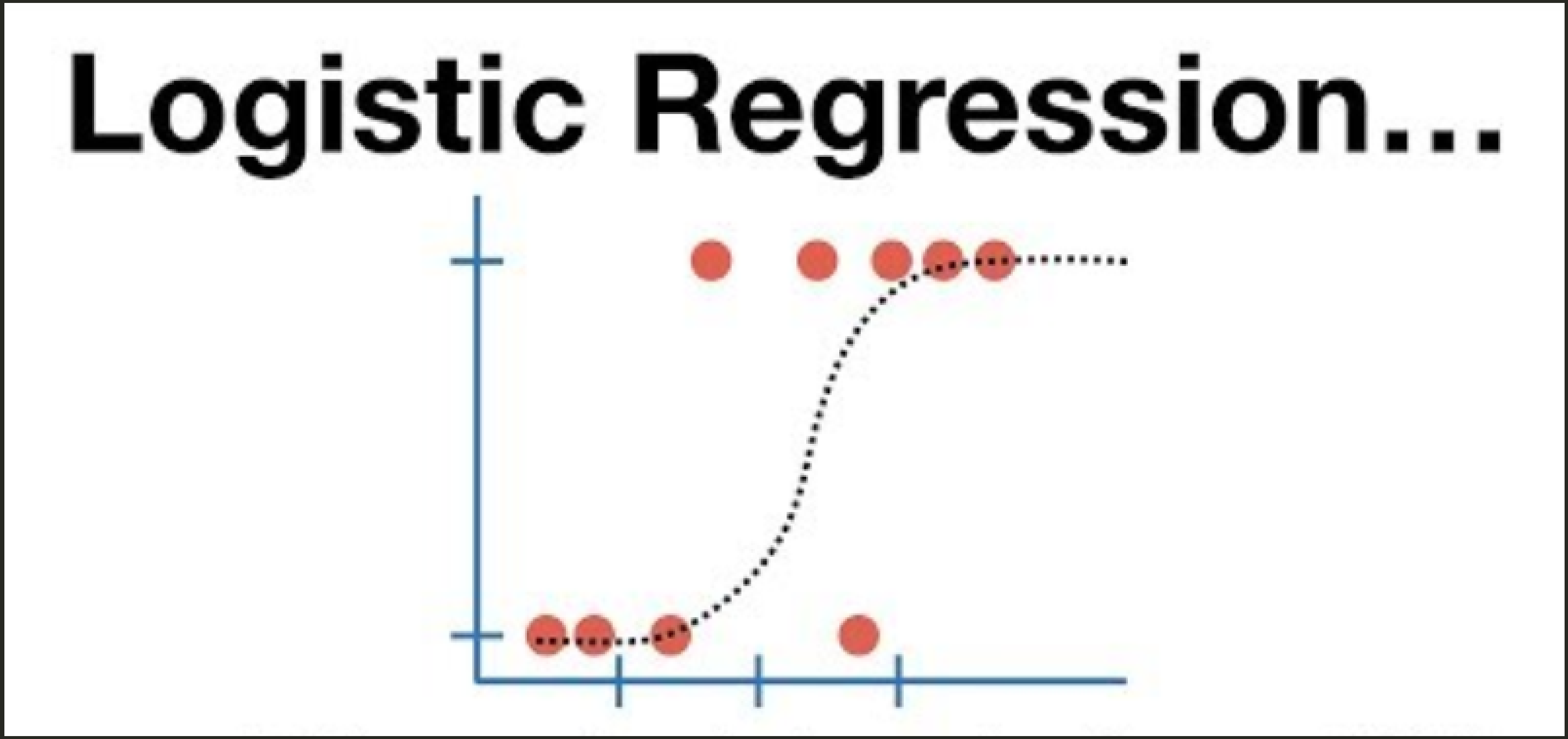
```
lm(Sale_Price ~ TotRms_AbvGrd, data = ames_data)$coefficients
```

```
##   (Intercept) TotRms_AbvGrd
##      18665.40      25163.83
```

- This is a common result when collinearity exists.
- Coefficients for correlated features become over-inflated and can fluctuate significantly.

Consequences

- One consequence of these large fluctuations in the coefficient terms is **overfitting**, which means we have high variance in the bias-variance tradeoff space.
- We can use tools such as **variance inflaction factors** (Myers, 1994) to identify and remove those strongly correlated variables, but it is not always clear which variable(s) to remove.
- Nor do we always wish to remove variables as this may be removing signal in our data.


Logistic Regression…

Agresti - Categorical Data Analysis (2002)

- Very intuitively written book
- Very detailed accompanying script by **Laura A. Thompson**
- The paper deals with categorical data analysis in general.

Faraway books on regression with R

- Logistic regression intuitively explained
- Examples with R-code
  - Faraway - **Extending the linear model with R**
  - Faraway - **Practical Regression and Anova using R**

Import the GESIS Panel dataset again

```
library(readstata13)
datf <- read.dta13("../data/ZA5666_v1-0-0_Stata14.dta",
                   convert.factors = F)
```

The argument `convert.factors`:

- logical. If TRUE, factors from Stata value labels are created.

A function to recode the missing values

```
transform_missings <- function(var){
  misvals <- c(-11,-22,-33,-44,-55,-66,-77,-88,-99,-111)
  var[var %in% misvals] <- NA
  return(var)
}
```

Variables for `glm`'s

- a11d056z: age group

```
table(datf$a11d056z)
```

```
##
## -99   1   2   3   4   5   6   7   8   9  10  11  12  13
##   5  31  87 101  91  83 100 163 159 133  64  56 105  44
```

```
age <- transform_missings(datf$a11d056z)
```

```
table(age)
```

```
## age
##   1   2   3   4   5   6   7   8   9  10  11  12  13
##  31  87 101  91  83 100 163 159 133  64  56 105  44
```

GP variable a11d094a: Children under 16 years

Does your household include children under 16?

- 1 Yes
- 2 No

```
children <- as.factor(transform_missings(datf$a11d094a))
table(children)
```

```
## children
##   1   2
## 325 681
```

Conditional Density Plot (GESIS Panel)

```
cdplot(children ~ age, data = dat)
```

Binary independent variables with `glm`

- The logistic regression belongs to the class of generalized linear models (GLM)
- The function for estimating a model of this class in is called `glm()`
- `glm()`

Specifying a `glm`

- formula object
- the class (binomial, gaussian, gamma)
- including link function (logit, probit, cauchit, log, cloglog)

must be specified

Logistic regression with R

```
glm_1 <- glm(children ~ age,
                    family = binomial())
```

```
sum_glm1 <- summary(glm_1)
sum_glm1$coefficients
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -0.7194058 0.16384386 -4.390801 1.129338e-05
## age          0.2225862 0.02376266  9.367056 7.458415e-21
```

Interpreting the coefficients

Consider the logistic model of children in household as a function of age.

```
sum_glm1$coefficients
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -0.7194058 0.16384386 -4.390801 1.129338e-05
## age          0.2225862 0.02376266  9.367056 7.458415e-21
```

- The estimates and standard errors are given in terms of log odds, not in terms of probability.
- The p-values mean the same thing they always have.

The inverse logit

```
sum_glm1$coefficients
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept) -0.7194058 0.16384386 -4.390801 1.129338e-05
## age          0.2225862 0.02376266  9.367056 7.458415e-21
```

- The coefficients can't be interpreted as simply as 'the children in household at age group 0'. We have to use the inverse logit in order to find that.

Log-odds of -0.7194058 is the same as probability 0.3275238.

```
library(faraway)
ilogit(sum_glm1$coefficients[1,1])
```

```
## [1] 0.3275238
```

About the intercept in a logistic model

- It is possible to get an intercept of less than 0.
- This means that the log-odds are negative, NOT the probability.
- E.g. a log-odds of 0 translates to a probability of 0.5.

Log-odds and the probability

- Log-odds always increases as probability increases.

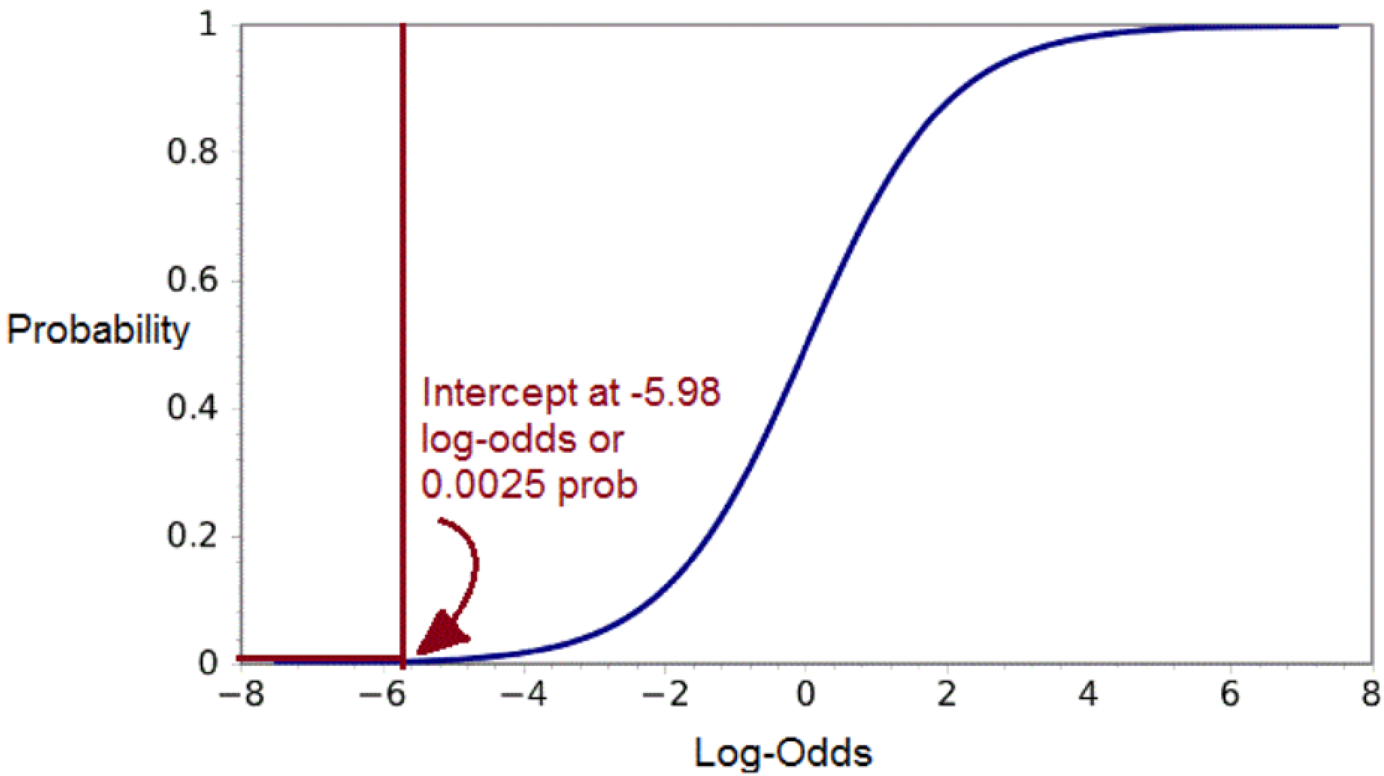
Therefore…

- A positive slope coefficient means that the response increases with the associated explanatory variable.

- In this case, the probability of children in the household increases with age.

Plotting the result

but it increases by the sigmoid curve, not at a constant rate.



Logistic regression model formula

Logistic models have regression formulas. This model's formula is:

Log-Odds(Children) = -0.7194058 + 0.2225862(Age) + error

We can plug age values into this formula to get predicted log-odds at different ages.

Log-odds for age group 5

-0.7194058 + 0.2225862*(5) = 0.3935251

Children probability in age group 5

```
ilogit(0.3935251)
```

```
## [1] 0.597131
```

Interpreting the results

- The difference between the null deviance and the residual deviance shows how our model is doing against the null model (a model with only the intercept). The wider this gap, the better.

```
anova(glm_1, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: children
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                  1000       1259
## age   1   98.956       999       1160 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Mc Fadden's R^2

```r
library(pscl)
pR2(glm_1)
```

```
##           llh       llhNull            G2      McFadden
## -580.02210772 -632.93066002  105.81710461    0.08359297
##          r2ML          r2CU
##    0.10031573    0.13978426
```

`llh`	The log-likelihood from the fitted model
`llhNull`	The log-likelihood from the intercept-only restricted model
`G2`	Minus two times the difference in the log-likelihoods
`McFadden`	McFadden's pseudo r-squared
`r2ML`	Maximum likelihood pseudo r-squared
`r2CU`	Cragg and Uhler's pseudo r-squared

Distance between residential area and large city

How far is it from where you live to the center of the nearest large city?

- 1 - In the center of a big city
- 6 - 60 km and more

```
region <- transform_missings(datf$bczd001a)
table(region)
```

```
## region
##   1   2   3   4   5   6
##  87 191 279 157 126 165
```

Satisfaction life in place of residence

How satisfied are you - all in all - with your life in [place of residence] at the moment?

- 1 - Very satisfied
- 5 - Very dissatisfied

```
satisfactionplace <- datf$a11c019a
table(satisfactionplace)
```

```
## satisfactionplace
##   1   2   3   4   5
## 553 534  99  30   6
```

Another model

```
glm_2 <- glm(children ~ age + satisfactionplace*region,
                    family = binomial())
```

```
pseudor2 <- pR2(glm_2)
pseudor2["McFadden"]
```

```
## McFadden
## 0.258121
```

Exercise: logistic regression with `Smarket` data

- load the `Smarket` data from the `ISLR` package
- create a `pairs` plot of the data
- check if there are missing values in the data
- have a look at the correltations between variables
- create a `corrplot`
- run a logistic model and look at the deviance

Another variable in the Gesis Panel data

- Number of tattoos:

```
Tatoos <- transform_missings(datf$bdao067a)
Tatoos[Tatoos==97]<-0
```

```
table(Tatoos)
```

```
## Tatoos
##   0   1   2   3   4   5   6
## 871  56  28  13   7   4   8
```

Generalized regression with R - more functions

- Logistic model with Probit link:

```
probitmod <- glm(children ~ age,
    family=binomial(link=probit))
```

- Regression with count data:

```
modp <- glm(Tatoos ~ age,family=poisson)
```

- Proportional odds logistic regression in library `MASS`:

```
library("MASS")
mod_plr<-polr(a11c020a ~ a11d096b ,data=dat)
```

Exercise: logistic regression

We will use a data on containing health-related measurements on women and whether they can be (or will be at a future point?) classified as diabetic. The data was collected by the US National Institute of Diabetes and is contained in the MASS package.

Load dataset and create test and train dataset

Load the `MASS` package and combine `Pima.tr` and `Pima.tr2` to a `data.frame` called train and save Pima.te as test. Change the coding of our variable of interest to (type) to 0 (non-diabetic) and 1 (diabetic). Check for and take note of any missing values.

Plotting using `pairs()` and `jitter`

Take a look at the data. Plot a scatterplot matrix between all the explanatory variables using `pairs()`, and color code the dots according to diabetic classification. Furthermore, try to plot type as a function of age. Use `jitter` to make your graph more informative. Bonus: Can you add a logistic fit based on age on top of your plot?

Exercise: logistic regression II

Coefficients and p-values

Using the `glm()` and the train data fit a logistic model of type on `age` and `bmi`. Print out the coefficients and their p-value.

Prediction

What does the model fitted in exercise 3 predict in terms of probability for someone age 35 with bmi of 32, what about bmi of 22?

Odds ratios

According to our model what are the odds that a woman in our sample is diabetic given age 55 and a bmi 37? Remember that odds in this context have a very precise definition which is different from probability.

Exercise: Confusion matrix

Build the confusion matrix, a table of actual diabetic classification against model prediction. Use a cutoff value of 0.5, meaning that women who the model estimates to have at least 0.5 chance of being diabetic are predicted to be diabetic. What is the prediction accuracy?

Links - regression

- Regression - **r-bloggers**
- The complete book of **Faraway**- very intuitive
- Good introduction on **Quick-R**
- **Multiple regression**
- **15 Types of Regression you should know**
- `ggeffects` **- Create Tidy Data Frames of Marginal Effects for 'ggplot' from Model Outputs**
- **Machine learning iteration**

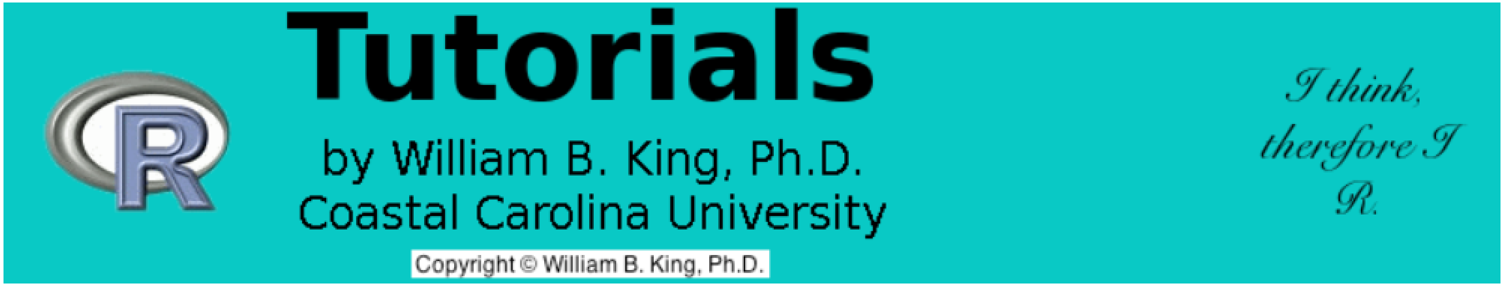
Nice table output with `stargazer`

```
library(stargazer)
stargazer(m3, type="html")
```

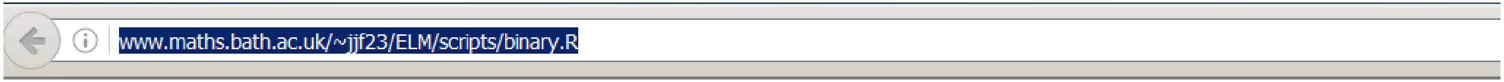
Example HTML output:

Links - logistic regression

- Introduction to **logistic regression**



- **Code for the book of Faraway**




```
library(faraway)
data(orings)
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim = c(0,1), xlab="Temperature", ylab="Prob of damage")
lmod <- lm(damage/6 ~ temp, orings)
abline(lmod)
logitmod <- glm(cbind(damage,6-damage) ~ temp, family=binomial, orings)
summary(logitmod)
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim = c(0,1), xlab="Temperature", ylab="Prob of damage")
```

- **Categorical data:** - **How to perform a Logistic Regression in R**

- Logistic Regression in R Tutorial

Links and resources

Shiny Apps - Diagnostics for linear regression

- **Simple Linear Regression**
- **Multicollinearity in multiple regression**
- Diagnostics for simple linear regression

Further resources

- Elegant regression results
- Regression analysis essentials