The Generalised Linear Model (2)

PSM 2

Bennett Kleinberg

22 Jan 2019

Welcome

Probability, Statistics & Modeling II

Lecture 3

GLM 2

What question do you have?

Today

- Recap linear regression
- Why the GLM?
- Extended cases: logistic regression
- How good is the model?
- How does one model compare to another?

Recap linear regression

Ingredients?

Recap linear regression

Core idea?

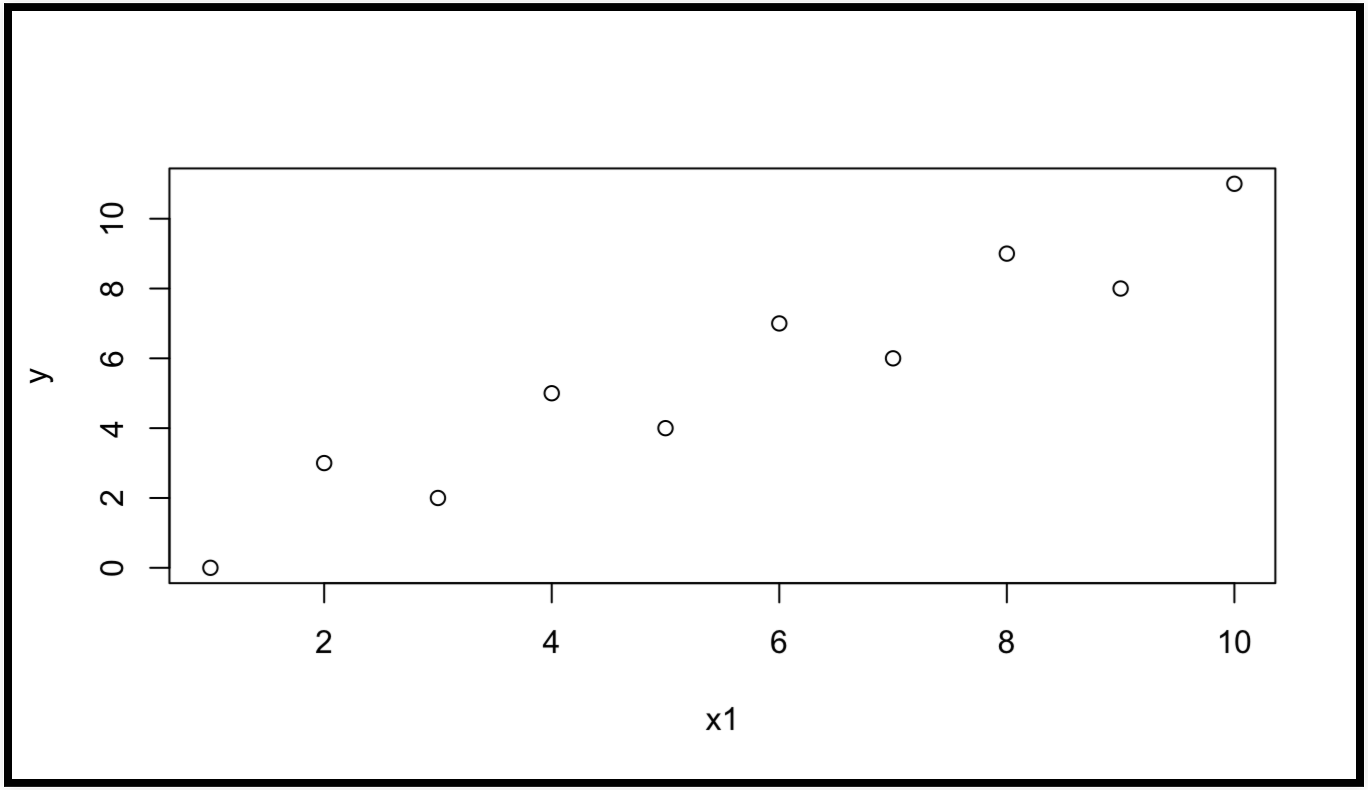
Recap linear regression

Types of effects?

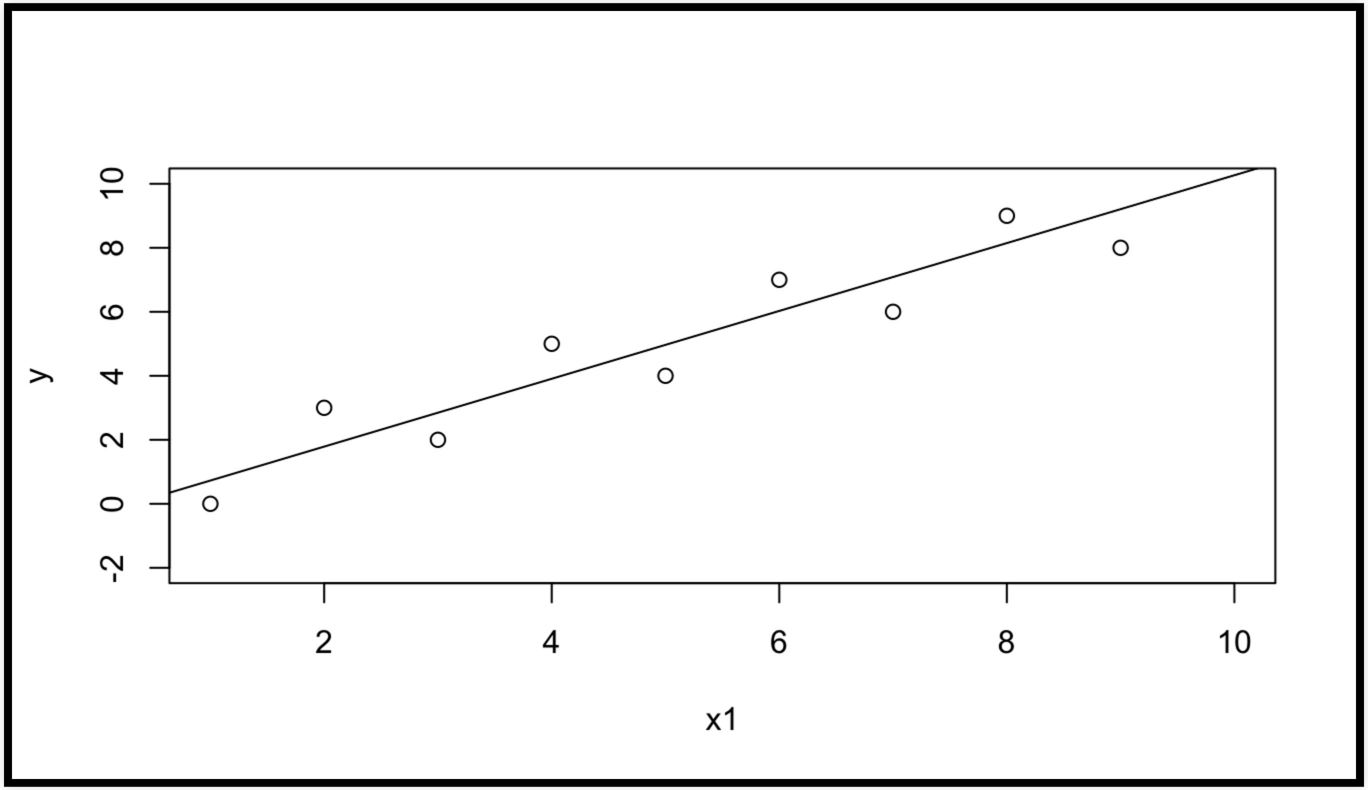
Recap linear regression

Residuals?

```r
x1 = 1:10
y = x1 + rep(c(-1, 1))
df = data.frame(x1, y)
plot(x1, y)
```



```
lm_1 = lm(y ~ x1, data=df)
{plot(x1, y, ylim=c(-2, 10))
  abline(lm_1)}
```



Continuation from last week

How to find the “optimal” terms for my model?

Maybe we can optimise this?

What if you don't know what the 'ideal' model is?

Especially neat for predictive modelling

Back to the shooting data:

```
load('./data/mass_shootings_detailed.RData')
smsd = smsd[smsd$school_related != 'Killed', ]
smsd = droplevels(smsd)
names(smsd)
```

```
##  [1] "caseid"         "n_fatal"         "n_injured"      "date"
##  [5] "day"            "age"             "gender"         "n_guns"
##  [9] "school_related" "mental_illness"
```

Automated variable selection

1. Specify the complete model

```
complete_model = lm(n_fatal ~ n_guns*mental_illness*school_related, data
```

2. Specify the null model

```
null_model = lm(n_fatal ~ 1, data = smsd)
```

3. Run model selection …

3 predictor variables: how many terms in the model?

- 1 intercept
- 3 main effects
- 3 2-way interactions
- 1 3-way interaction

Model selection

```r
summary(complete_model)
```

```
##
## Call:
## lm(formula = n_fatal ~ n_guns * mental_illness * school_related,
##     data = smsd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9592 -2.1233 -0.6777  1.2421 26.2074
##
## Coefficients:
##                                            Estimate Std. Error t value
## (Intercept)                                 2.30041    0.93210   2.468
## n_guns                                      0.86436    0.39577   2.184
## mental_illnessYes                           1.47991    1.28991   1.147
## school_relatedYes                          -0.01274    1.70127  -0.007
## n_guns:mental_illnessYes                    0.03300    0.49495   0.067
## n_guns:school_relatedYes                   -1.02874    0.77367  -1.330
## mental_illnessYes:school_relatedYes        -3.41734    2.24208  -1.524
```

Model selection

```
summary(null_model)
```

```
##
## Call:
## lm(formula = n_fatal ~ 1, data = smsd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4751 -2.4751 -0.4751  1.5249 27.5249
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.4751     0.3338    13.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.491 on 180 degrees of freedom
```

Model selection: backward

```r
step(complete_model, direction = 'backward')
```

```
## Start:  AIC=511.13
## n_fatal ~ n_guns * mental_illness * school_related
##
##                                        Df Sum of Sq    RSS    AIC
## <none>                                              2790.6 511.13
## - n_guns:mental_illness:school_related  1    72.456 2863.0 513.77
```

```
##
## Call:
## lm(formula = n_fatal ~ n_guns * mental_illness * school_related,
##     data = smsd)
##
## Coefficients:
##                               (Intercept)
##                                   2.30041
##                                    n_guns
##                                   0.86436
##                         mental_illnessYes
##                                   1.47991
##                         school_relatedYes
##                                  -0.01274
##                  n_guns:mental_illnessYes
##                                   0.03300
##                  n_guns:school_relatedYes
##                                  -1.02874
```

Model selection: forward

```r
step(null_model, direction = 'forward'
     , scope=list(lower=null_model, upper=complete_model))
```

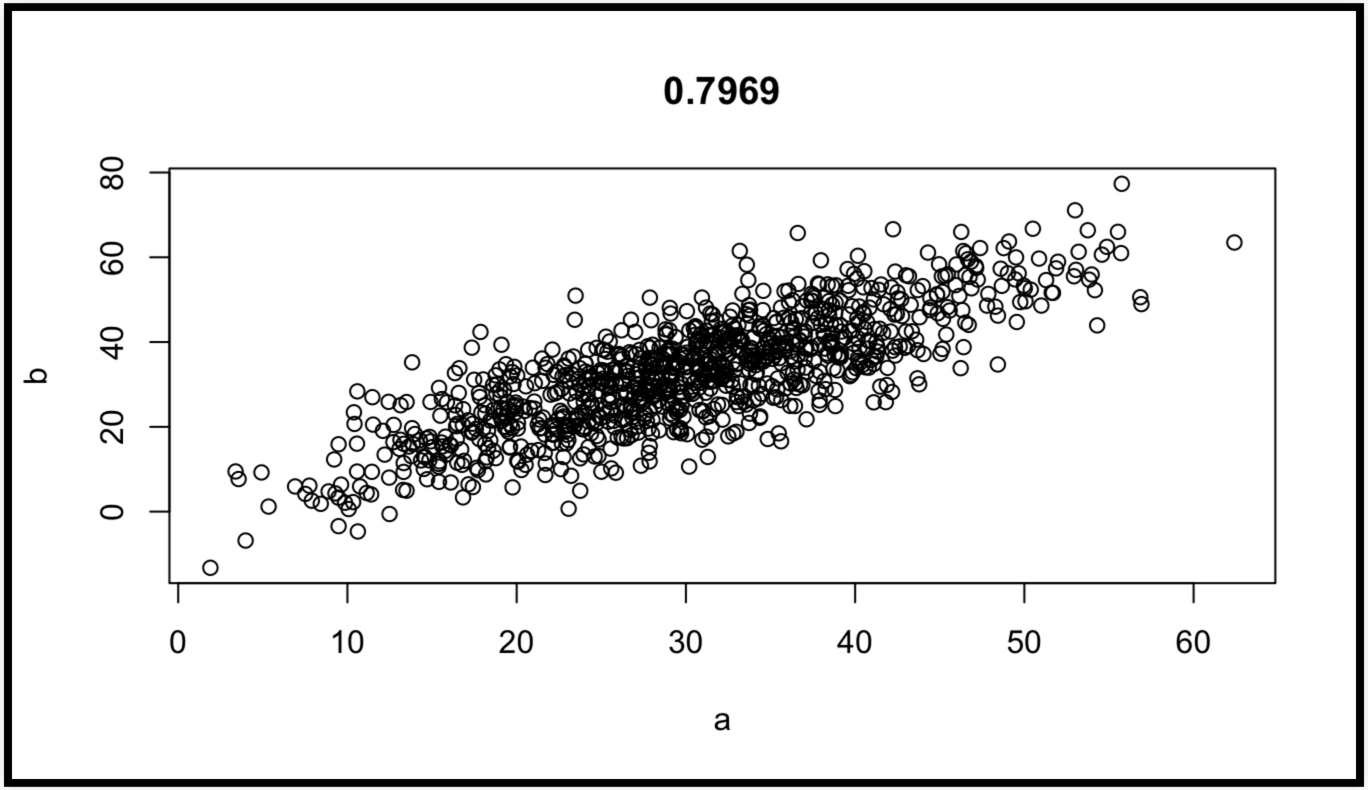
```
## Start:  AIC=544.78
## n_fatal ~ 1
##
##                   Df Sum of Sq    RSS    AIC
## + n_guns           1    535.46 3095.7 517.91
## + mental_illness   1    174.44 3456.7 537.87
## + school_related   1     55.94 3575.2 543.97
## <none>                         3631.1 544.78
##
## Step:  AIC=517.91
## n_fatal ~ n_guns
##
##                   Df Sum of Sq    RSS    AIC
## + mental_illness   1    95.460 3000.2 514.24
## + school_related   1    57.394 3038.3 516.52
## <none>                         3095.7 517.91
##
## Step:  AIC=514.24
```

```
##
## Call:
## lm(formula = n_fatal ~ n_guns + mental_illness + school_related +
##     n_guns:mental_illness, data = smsd)
```

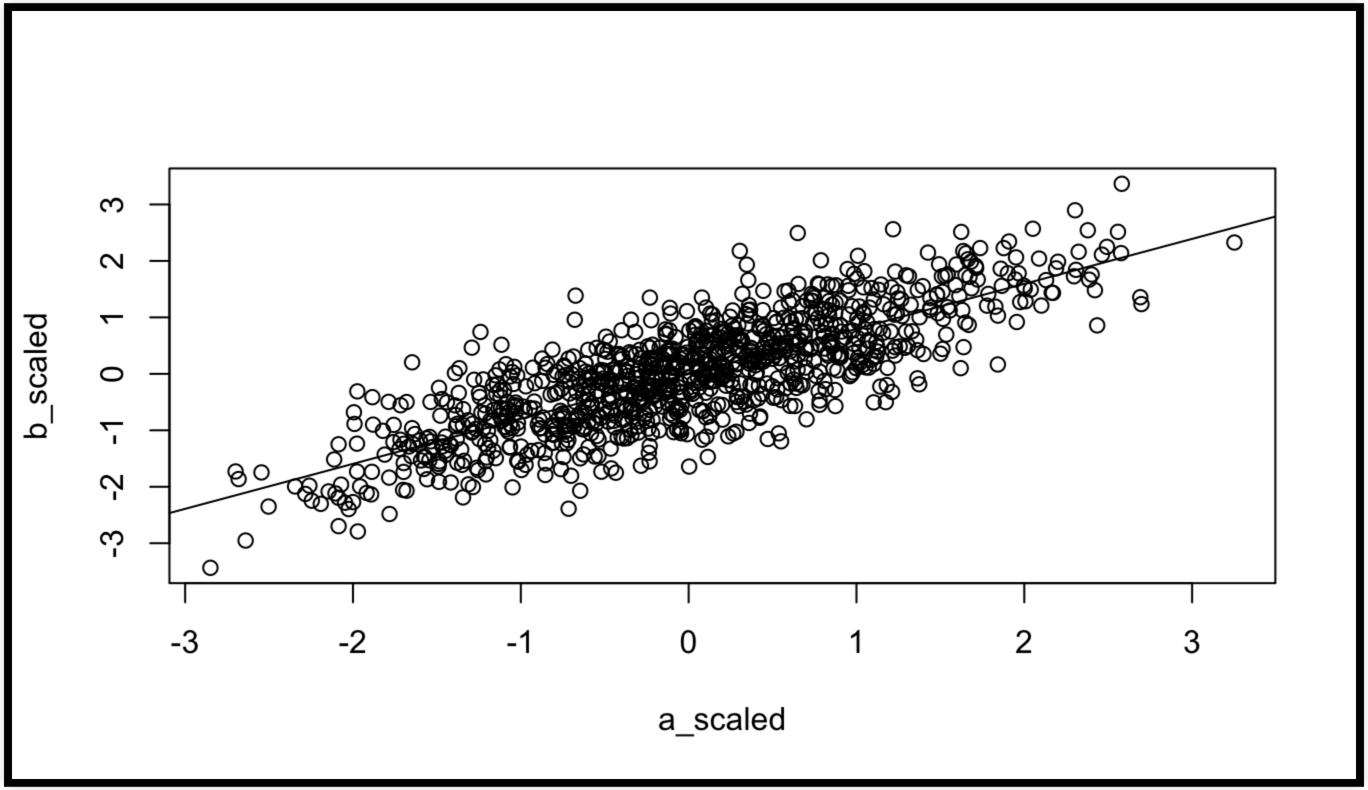
```
##

## Coefficients:
##              (Intercept)                       n_guns
##                   2.7122                       0.6060
##         mental_illnessYes             school_relatedYes
##                   0.3975                      -1.5038
## n_guns:mental_illnessYes
##                   0.6247
```

Limitations of linear regression?

```
set.seed(123)
a = rnorm(1000, 30, 10)
b = a + rnorm(1000, 2, 8)
plot(a, b, main = round(cor(a, b), 4))
```



```r
a_scaled = scale(a)
b_scaled = scale(b)
{plot(a_scaled, b_scaled)
abline(lm(a_scaled ~ b_scaled))}
```



```r
lm(a_scaled ~ b_scaled - 1)
```

```
##
## Call:
## lm(formula = a_scaled ~ b_scaled - 1)
##
## Coefficients:
## b_scaled
##   0.7969
```

Limitations of linear regression?

- Correlation != causation
- **Continuous outcome variable**

Generalising the model

The Generalised Linear Model

GLM in general

- framework to deal with different outcome variables
- uses the same "linearity in parameters" idea
- key feature: linking the outcome to the predictor(s)

The GLM in R

```
my_model_glm = glm(formula = n_fatal ~ n_guns*mental_illness*school_relat
                  , family = gaussian
                  , data = smsd)
summary(my_model_glm)
```

```
##
## Call:
## glm(formula = n_fatal ~ n_guns * mental_illness * school_related,
##     family = gaussian, data = smsd)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.9592  -2.1233  -0.6777   1.2421  26.2074
##
## Coefficients:
##                                       Estimate Std. Error t value
## (Intercept)                            2.30041    0.93210   2.468
## n_guns                                 0.86436    0.39577   2.184
## mental_illnessYes                      1.47991    1.28991   1.147
## school_relatedYes                     -0.01274    1.70127  -0.007
## n_guns:mental_illnessYes               0.03300    0.49495   0.067
## n_guns:school_relatedYes              -1.02874    0.77367  -1.330
## mental_illnessYes:school_relatedYes   -3.41734    2.24208  -1.524
```

Compared to `lm`

```
my_model_lm = lm(formula = n_fatal ~ n_guns*mental_illness*school_relatec
                 , data = smsd)
summary(my_model_lm)
```

```
##
## Call:
## lm(formula = n_fatal ~ n_guns * mental_illness * school_related,
##     data = smsd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9592 -2.1233 -0.6777  1.2421 26.2074
##
## Coefficients:
##                                            Estimate Std. Error t value
## (Intercept)                                 2.30041    0.93210   2.468
## n_guns                                      0.86436    0.39577   2.184
## mental_illnessYes                           1.47991    1.28991   1.147
## school_relatedYes                          -0.01274    1.70127  -0.007
## n_guns:mental_illnessYes                    0.03300    0.49495   0.067
## n_guns:school_relatedYes                   -1.02874    0.77367  -1.330
## mental_illnessYes:school_relatedYes        -3.41734    2.24208  -1.524
```

GLM vs LM

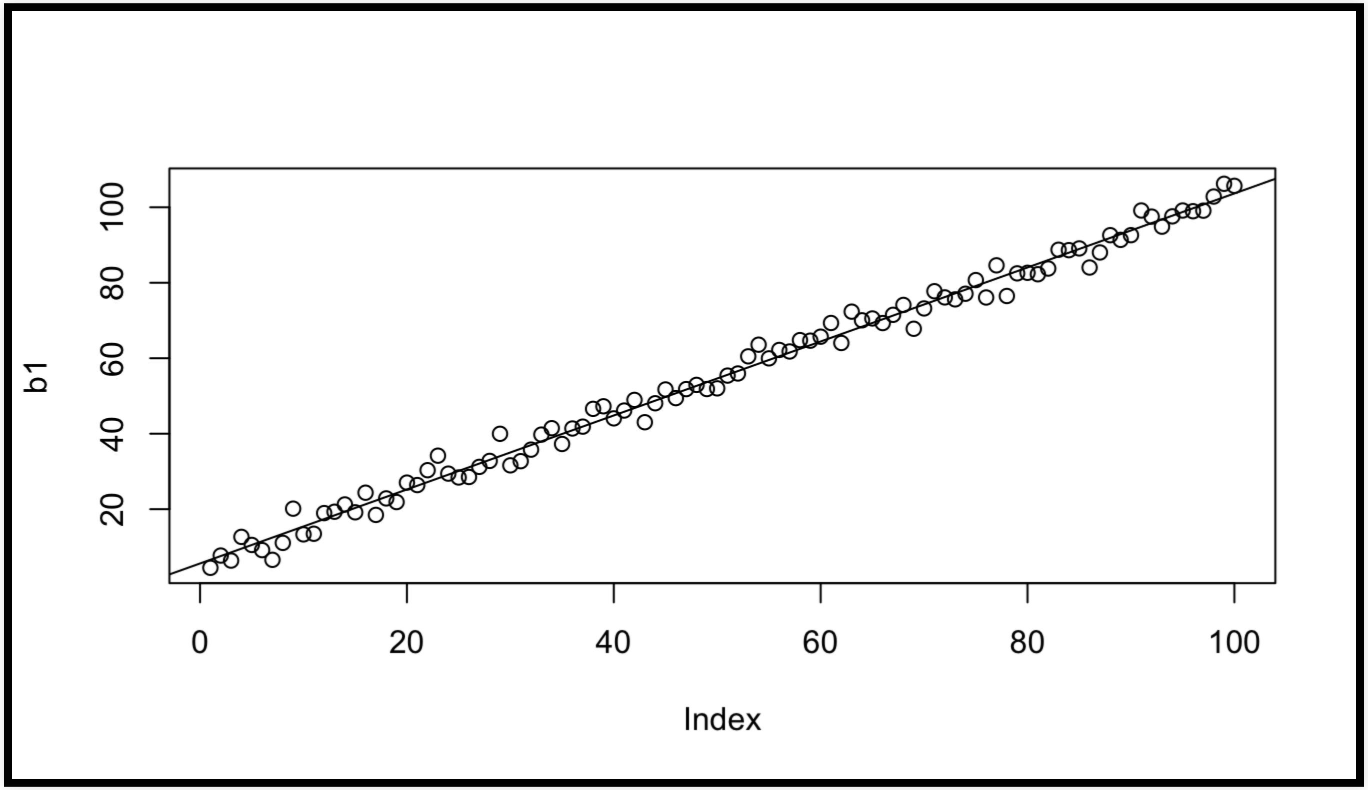
The LM is a GLM with the Gaussian link function.

- link function ‘links’ the linear predictor to the mean of the distribution of the outcome variable
- e.g. if outcome variable from normal distribution –> “normal” link function (Gaussian)
- e.g. if outcome variable from poisson distribution –> “Poisson” link (Log)
- e.g. if outcome variable from binomial distributiob –> “Binomial” link (Logit)

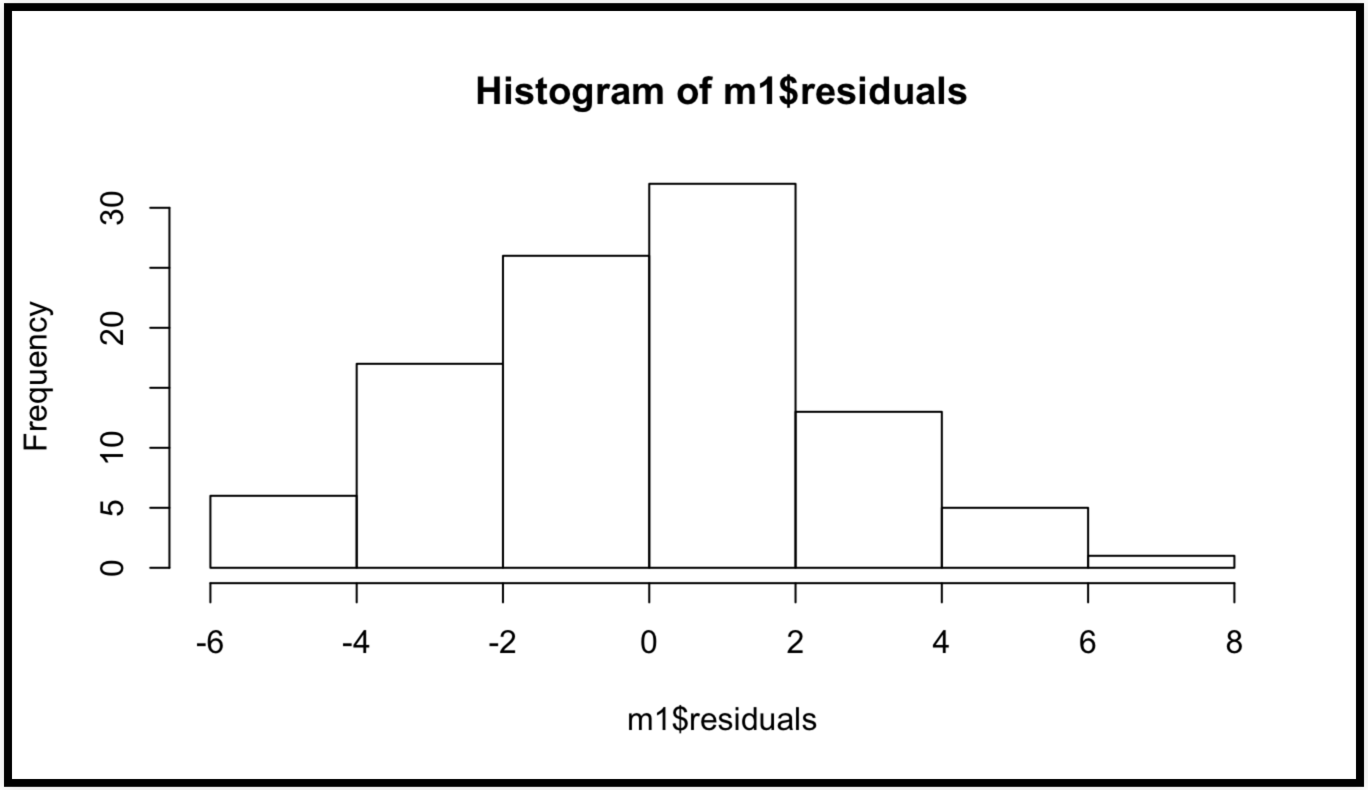
Why bother with this?

Compare:

```
b1 = 1:100 + rnorm(100, 5, 3)
df1 = data.frame(a1 = 1:100, b1)
{plot(b1)
  abline(lm(b1 ~a1, data=df1))}
```



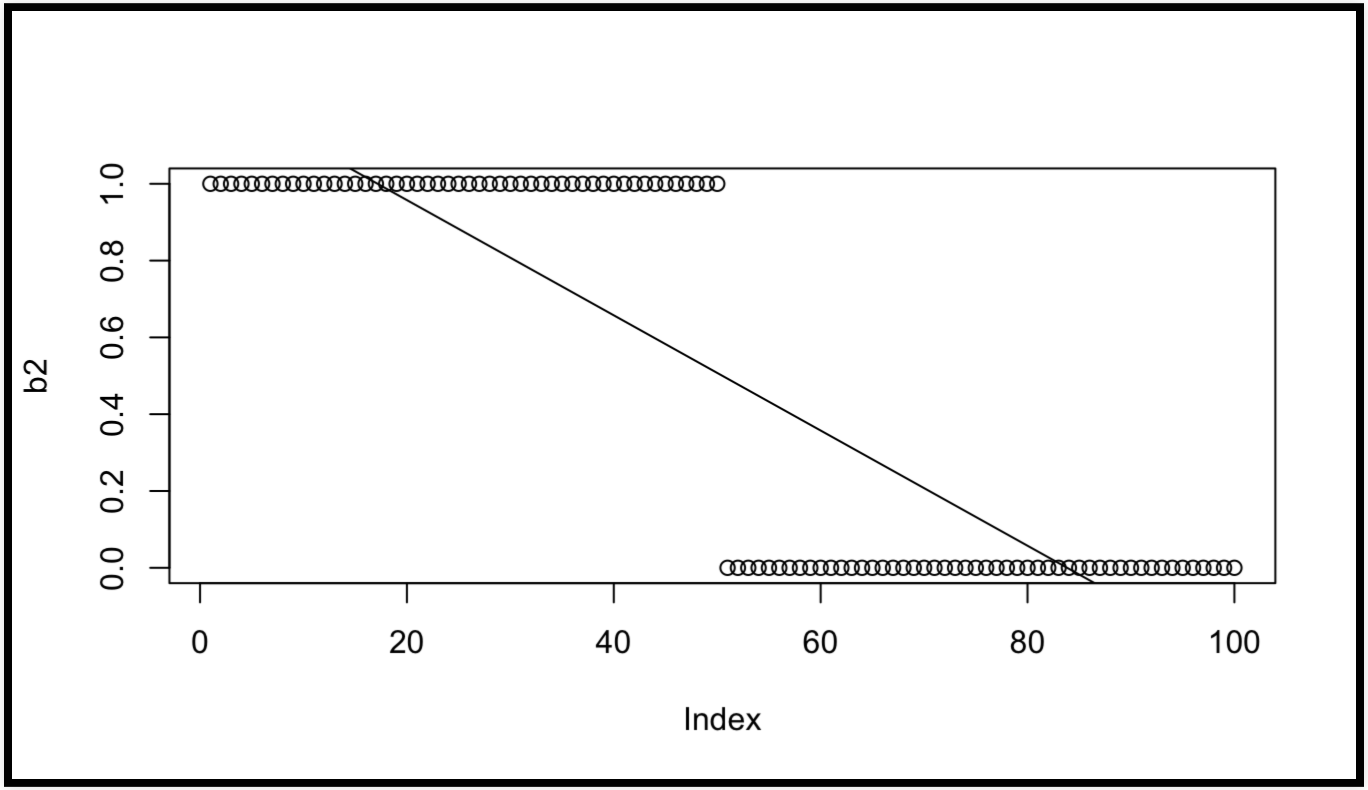
```r
m1 = lm(b1 ~a1, data=df1)
hist(m1$residuals)
```



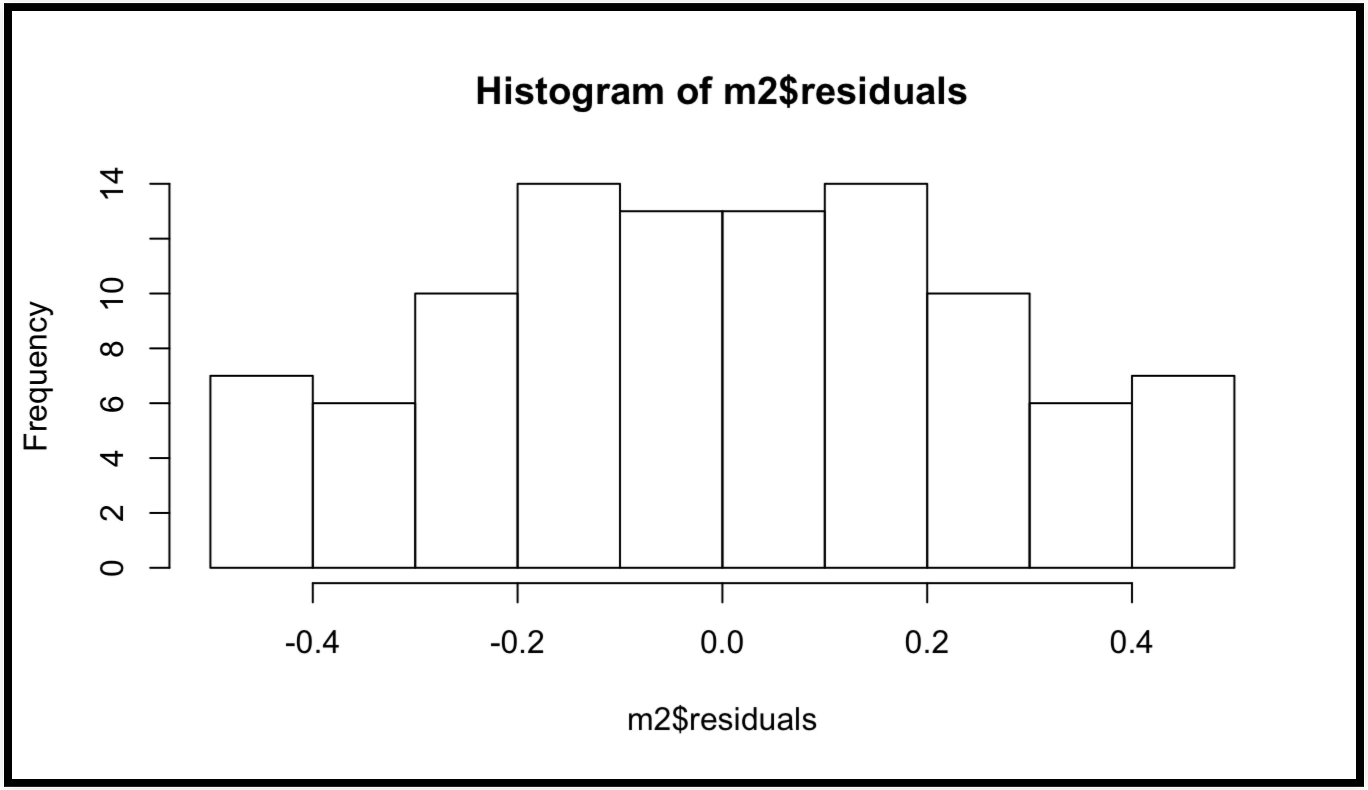
Why bother with this?

Compare:

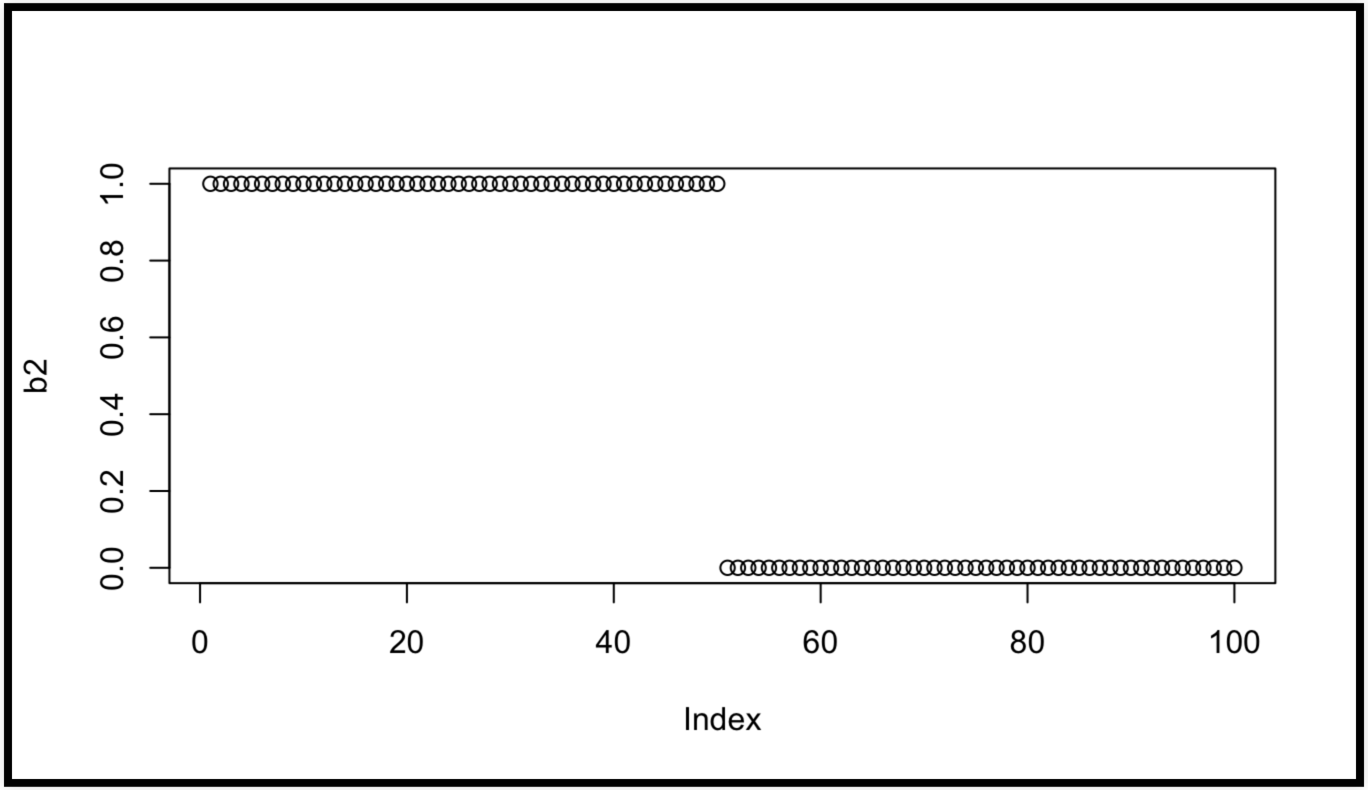
```
b2 = rep(c(1,0), each=50)
df2 = data.frame(a2 = 1:100, b2)
{plot(b2)
  abline(lm(b2 ~a2, data=df2))}
```



```
m2 = lm(b2 ~a2, data=df2)
hist(m2$residuals)
```



```r
plot(b2)
```



What to do?

We need a representation of the outcome variable…

- that is linear to the predictor
- i.e. transforms the data to so that Y has a linear relationship to the predictors

But which function does this?

The link function

Answer: for binary outcomes, the **logit** function

- transforms the outcome to a continuous probability
- and uses the log-odds to model a linear relationship between X and Y

The logit function

- maps `0,1` values to `-Inf : Inf`
- assumes a probability of `P(Y == 1)`
- probability is expressed as the odds
- linearity through the log of the odds

```
prob = 0.40
odds = prob/(1-prob)
odds
```

```
## [1] 0.6666667
```

```
log(odds)
```

```
## [1] -0.4054651
```

Intermezzo: odds

	Smoker	Nonsmoker
Dead	30	20
Alive	70	80
	100	100

```
#Odds of smoker dead:
(30/100)/(70/100)
```

```
## [1] 0.4285714
```

```
# equal to
30/70
```

```
## [1] 0.4285714
```

Intermezzo: odds

Odds = event_present/event_not_present

->

```
odds = P/(1-P)
```

Intermezzo: odds

	Smoker	**Nonsmoker**
Dead	30	20
Alive	70	80
	100	100

Odds of nonsmoker alive?

Intermezzo: odds

	Smoker	Nonsmoker
Dead	30	20
Alive	70	80
	100	100

```
#Odds of nonsmoker alive:
80/20
```

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

Intermezzo: odds

	Smoker	**Nonsmoker**
Dead	30	20
Alive	70	80
	100	100

Odds of nonsmoker ~~alive~~ dead?

Intermezzo: odds

	Smoker	**Nonsmoker**
Dead	30	20
Alive	70	80
	100	100

Odds of nonsmoker ~~alive~~ dead?

```
20/80
```

```
## [1] 0.25
```

```
1/(80/20)
```

```
## [1] 0.25
```

Intermezzo: odds

	Smoker	Nonsmoker
Dead	30	20
Alive	70	80
	100	100

Odds ratio: association between both factors.

```
OR = (30/70)/(20/80)
OR
```

```
## [1] 1.714286
```

The logit function

Models the binary outcome through the log odds of the predictors.

p	odds	logodds
.001	.001001	-6.906755
.01	.010101	-4.59512
.15	.1764706	-1.734601
.2	.25	-1.386294
.25	.3333333	-1.098612
.3	.4285714	-.8472978
.35	.5384616	-.6190392
.4	.6666667	-.4054651
.45	.8181818	-.2006707
.5	1	0
.55	1.222222	.2006707
.6	1.5	.4054651
.65	1.857143	.6190392
.7	2.333333	.8472978
.75	3	1.098612

.75	3	1.098612
.8	4	1.386294
.85	5.666667	1.734601
.9	9	2.197225
.999	999	6.906755
.9999	9999	9.21024

Implication: transformation of the coefficients

Remeber: we model the log of the odds ratio.

The logit function

Implication: transformation of the coefficients

Remeber: we model the log of the odds ratio.

So we need to ‘unlog’ the coefficients to get the odds.

Case today: Parole data

Dataset from Kaggle.

```
load('./data/parole_data.RData')
parole_data
```

```
##            sex     race granted
## 1         MALE    WHITE       1
## 2         MALE HISPANIC       0
## 3         MALE    BLACK       0
## 4         MALE    BLACK       1
## 5         MALE HISPANIC       0
## 6         MALE HISPANIC       1
## 7         MALE    BLACK       0
## 8         MALE    WHITE       0
## 9         MALE    WHITE       0
## 10        MALE HISPANIC       0
## 11        MALE    WHITE       1
## 12        MALE    BLACK       1
## 13        MALE    BLACK       0
## 14        MALE HISPANIC       1
## 15        MALE    WHITE       0
## 16        MALE    BLACK       0
## 17        MALE    BLACK       0
```

Suppose we model the success of parole hearings…

```r
parole_success = glm(granted ~ sex
                     , data=parole_data
                     , family = 'binomial')
parole_success
```

```
##
## Call:  glm(formula = granted ~ sex, family = "binomial", data = parole
##
## Coefficients:
## (Intercept)      sexMALE
##       2.392       -1.363
##
## Degrees of Freedom: 41534 Total (i.e. Null);  41533 Residual
## Null Deviance:        46850
## Residual Deviance: 46320      AIC: 46320
```

```r
summary(parole_success)
```

```
##
## Call:
## glm(formula = granted ~ sex, family = "binomial", data = parole_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2268  -1.6337   0.7818   0.7818   0.7818
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.39190    0.06916   34.58   <2e-16 ***
## sexMALE     -1.36305    0.07011  -19.44   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 46854  on 41534  degrees of freedom
```

What does this mean?

```r
coefficients(parole_success)
```

```
## (Intercept)     sexMALE
##    2.391896   -1.363046
```

What would the lm interpretation be?

Coefficient interpretation

Remember what the logit function does?

```
Y ~ log_odds_ratio(X)
```

… So the coefficient `X` needs to be transformed.

Coefficient interpretation

Transforming the coefficient:

log odds to **probability**

- **log** –> un-log
  - natural logarithm reverse
  - `e` —> `exp()` in R

Coefficient interpretation

Let’s use the output and transform:

```
coefficients(parole_success)
```

```
## (Intercept)     sexMALE
##    2.391896   -1.363046
```

```
exp(-1.36)
```

```
## [1] 0.2566608
```

Understanding the odds

```
table(parole_data$granted, parole_data$sex)
```

```
##
##        FEMALE  MALE
##    0      228 10220
##    1     2493 28594
```

Odds by hand

	Female	Male
0	228	10220
1	2493	28594
	2721	38814

Odds of male granted:

```
(28594/38814)/(10220/38814)
```

```
## [1] 2.797847
```

```
#Note: equivalent to 28594/10220
```

Odds of female granted:

```
2493/228
```

```
## [1] 10.93421
```

Odds ratio male to female granted

```
2.7978/10.9342
```

```
## [1] 0.2558761
```

Proof:

```
log(0.2558)
```

```
## [1] -1.363359
```

```
coefficients(parole_success)
```

```
## (Intercept)     sexMALE
##    2.391896   -1.363046
```

Interpretation

Conversely: Female to male odds ration…

```
10.9342/2.7978
```

```
## [1] 3.908142
```

The odds of being granted parole as a female are 3.90 times the odds of being granted parole as a male.

Add additional factor?

```
tapply(parole_data$granted, list(parole_data$race), mean)
```

```
##     BLACK  HISPANIC     WHITE
## 0.6965374 0.7137377 0.8355549
```

Extend the model

```r
parole_success_2 = glm(granted ~ sex + race
                     , data=parole_data
                     , family = 'binomial')
summary(parole_success_2)
```

```
##
## Call:
## glm(formula = granted ~ sex + race, family = "binomial", data = parole
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3921  -1.5223   0.6224   0.8680   0.8680
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.04089    0.07057  28.920  < 2e-16 ***
## sexMALE      -1.25900    0.07054 -17.848  < 2e-16 ***
## raceHISPANIC  0.09646    0.02909   3.317 0.000911 ***
## raceWHITE     0.76131    0.02763  27.557  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

Interpretation

```
coefficients(parole_success_2)
```

```
##   (Intercept)       sexMALE raceHISPANIC    raceWHITE
##    2.04089495   -1.25900315   0.09646174   0.76131449
```

???

–> Key: odds ratio to reference group

Interpretation

```
# --> sex: MALE to FEMALE
exp(-1.259)
```

```
## [1] 0.2839378
```

```
# --> race: HISPANIC to BLACK
exp(0.096)
```

```
## [1] 1.100759
```

```
# --> race: WHITE to BLACK
exp(0.7613)
```

```
## [1] 2.141058
```

Adding more...

```
interaction.plot(parole_data$sex, parole_data$race, parole_data$granted)
```

Interactions?

```
tapply(parole_data$granted, list(parole_data$sex, parole_data$race), mear
```

```
##               BLACK  HISPANIC     WHITE
## FEMALE 0.8870804 0.8757764 0.9456215
## MALE   0.6859737 0.7072319 0.8236240
```

Extend the model further

```
parole_success_3 = glm(granted ~ sex*race
                       , data=parole_data
                       , family = 'binomial')
parole_success_3
```

```
##
## Call:  glm(formula = granted ~ sex * race, family = "binomial", data =
##
## Coefficients:
##           (Intercept)                sexMALE           raceHISPANIC
##               2.06126               -1.27990               -0.10823
##             raceWHITE   sexMALE:raceHISPANIC      sexMALE:raceWHITE
##               0.79461                0.20885               -0.03488
##
## Degrees of Freedom: 41534 Total (i.e. Null);  41529 Residual
## Null Deviance:       46850
## Residual Deviance: 45470     AIC: 45480
```

Interpretation

```
coefficients(parole_success_3)
```

```
##          (Intercept)              sexMALE          raceHISPANIC
##           2.06125922          -1.27989659           -0.10823161
##            raceWHITE sexMALE:raceHISPANIC     sexMALE:raceWHITE
##           0.79461370           0.20884686           -0.03488046
```

Have a look at this Stackexchange answer.

Interpretation

```
exp(coefficients(parole_success_3))
```

```
##         (Intercept)             sexMALE        raceHISPANIC
##           7.8558559           0.2780661           0.8974197
##           raceWHITE sexMALE:raceHISPANIC   sexMALE:raceWHITE
##           2.2135857           1.2322563           0.9657208
```

Odds ratios!

> *For males the OR of BLACK to HISPANIC is 1.23 the OR of females*


```
exp(coefficients(parole_success_3))
```

```
##         (Intercept)             sexMALE        raceHISPANIC
##           7.8558559           0.2780661           0.8974197
##           raceWHITE sexMALE:raceHISPANIC   sexMALE:raceWHITE
##           2.2135857           1.2322563           0.9657208
```

Odds ratios!

For males the OR of BLACK to WHITE is 0.97 the OR of females

```
exp(coefficients(parole_success_3))
```

```
##         (Intercept)              sexMALE         raceHISPANIC
##           7.8558559            0.2780661            0.8974197
##           raceWHITE sexMALE:raceHISPANIC    sexMALE:raceWHITE
##           2.2135857            1.2322563            0.9657208
```

Odds ratios!

For males the OR of granted parole is 0.27 the OR of females


```
exp(coefficients(parole_success_3))
```

```
##          (Intercept)               sexMALE          raceHISPANIC
##            7.8558559             0.2780661             0.8974197
##            raceWHITE sexMALE:raceHISPANIC     sexMALE:raceWHITE
##            2.2135857             1.2322563             0.9657208
```

Odds ratios!

For WHITE defendants the OR of granted parole is 2.21 the OR of BLACK defendants

Connections to machine learning

- Regression the best starting point
- Core difference: explanatory modelling vs predictive modelling
- More care against overfitting in predictive modelling
- Split the data

Goodness-of-fit of a model

Assessing how good a model is

Model fit

Model fit

Explained variance: R-squared (multiple vs adjusted)

```r
summary(complete_model)
```

```
##
## Call:
## lm(formula = n_fatal ~ n_guns * mental_illness * school_related,
##     data = smsd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9592 -2.1233 -0.6777  1.2421 26.2074
##
## Coefficients:
##                                          Estimate Std. Error t value
## (Intercept)                               2.30041    0.93210   2.46
## n_guns                                    0.86436    0.39577   2.18
## mental_illnessYes                         1.47991    1.28991   1.14
## school_relatedYes                        -0.01274    1.70127  -0.00
## n_guns:mental_illnessYes                  0.03300    0.49495   0.06
## n_guns:school_relatedYes                 -1.02874    0.77367  -1.33
## mental_illnessYes:school_relatedYes      -3.41734    2.24208  -1.52
```

Model fit

Mean squared error

```r
mean(complete_model$residuals^2)
```

```
## [1] 15.41751
```

Model fit

Root mean square error

```
sqrt(mean(complete_model$residuals^2))
```

```
## [1] 3.926514
```

Model fit

Mean absolute error

```
mean(abs(complete_model$residuals))
```

```
## [1] 2.519573
```

Model fit

Mean percentage error

```
mean(complete_model$residuals/(complete_model$model$n_fatal+1)*100)
```

```
## [1] -50.7433
```

Model fit

Mean absolute percentage error

```
mean(abs(complete_model$residuals/(complete_model$model$n_fatal+1))*100)
```

```
## [1] 75.17279
```

Model comparison

When to choose one model over the other?

Model comparison

Idea: 2 models compete

Requirement: the two models are nested

Nested models

```
model_1 = lm(n_fatal ~ mental_illness, data = smsd)
model_2 = lm(n_fatal ~ mental_illness+school_related, data = smsd)
model_3 = lm(n_fatal ~ mental_illness*school_related, data = smsd)
```

Rough model evaluation

```
sqrt(mean(model_1$residuals^2))
```

```
## [1] 4.370098
```

```
sqrt(mean(model_2$residuals^2))
```

```
## [1] 4.311128
```

```
sqrt(mean(model_3$residuals^2))
```

```
## [1] 4.306262
```

But you want to be precise…

Model comparison test

```
anova(model_1, model_2)
```

```
## Analysis of Variance Table
##
## Model 1: n_fatal ~ mental_illness
## Model 2: n_fatal ~ mental_illness + school_related
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    179 3456.7
## 2    178 3364.0  1     92.66 4.9029 0.02808 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

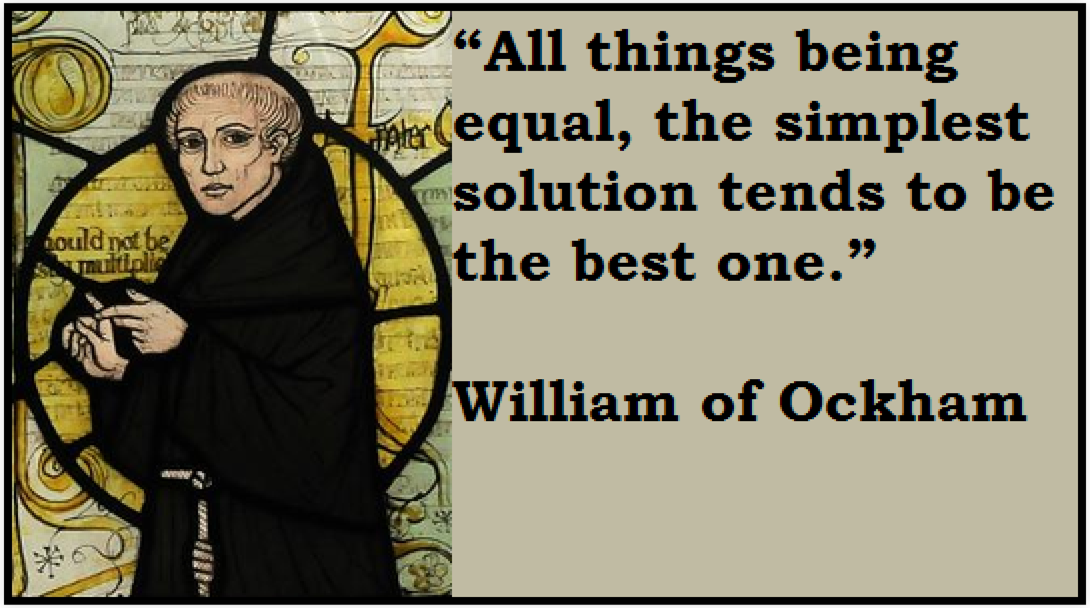
```r
anova(model_2, model_3)
```

```
## Analysis of Variance Table
##
## Model 1: n_fatal ~ mental_illness + school_related
## Model 2: n_fatal ~ mental_illness * school_related
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    178 3364.0
## 2    177 3356.4  1     7.589 0.4002 0.5278
```

```r
anova(model_1, model_3)
```

```
## Analysis of Variance Table
##
## Model 1: n_fatal ~ mental_illness
## Model 2: n_fatal ~ mental_illness * school_related
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)  
## 1    179 3456.7                              
## 2    177 3356.4  2    100.25 2.6433 0.07393 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model comparison



Model comparison

- only if additional parameters improve the model significantly
- vice versa: you only reject your model if it’s significantly worse than a more complicated model

RECAP

- model selection
- logistic regression
- coefficient interpretation
- model selection

Outlook

Next week

- Hypothesis testing beyond t-tests
- GLM as ANOVA

Homework

- Advanced regression modelling in R

END