# *Logistic and binomial regression*

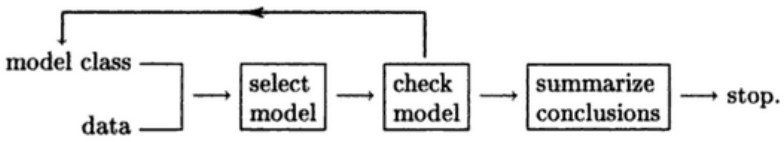
*Ben Bolker*

*October 3, 2018*

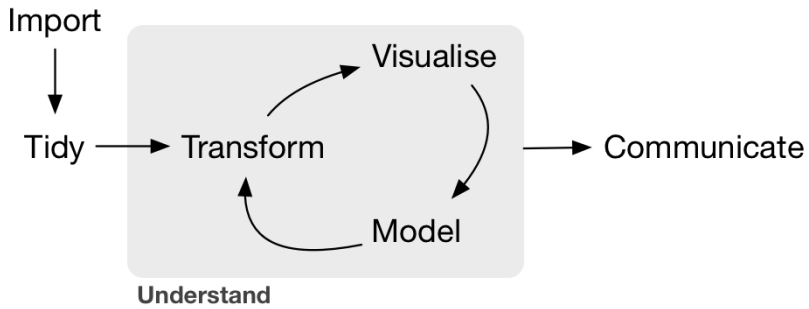


## *modeling*

### *data analysis road map*




(McCullagh and Nelder, 1989)




from Hadley Wickham

(https://jules32.github.io/2016-07-12-Oxford/dplyr_tidyr/)

These are good, but they don't address the **data snooping** problem.

1. figure out the (subject-area) question
2. design experiment/data collection (power analysis; simulation)

3. *collect data*

4. understand the data
5. specify the model; *write it down!*

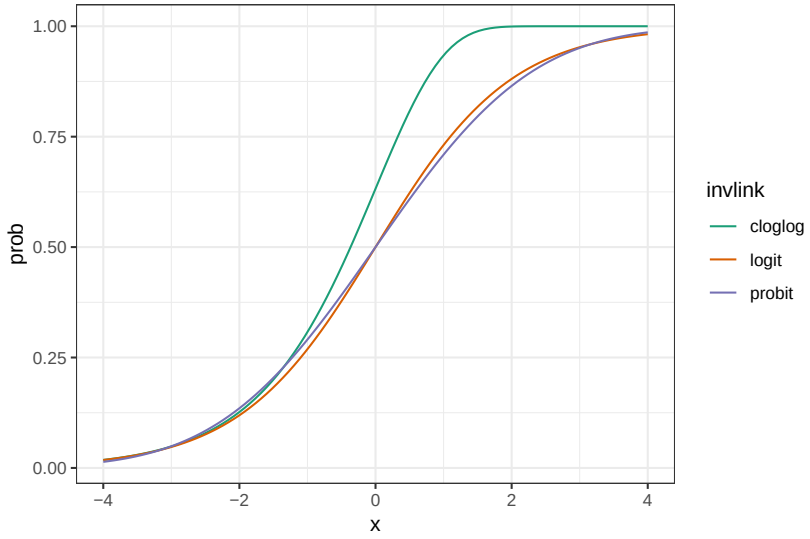
6. inspect data (Q/A) (return to 5?  )
7. fit model
8. graphical & quantitative diagnostics (return to 5? )
9. interpret parameters; inference; plot results

## *basics*

Can use *any* smooth function from $(0,1) \to \mathbb{R}$ as the link function

- *logistic* regression: binary data with a logit link (inverse-link=logistic)
- *binomial* (or *aggregated binomial* regression: binomial data (maybe logit link, maybe other)
- *probit regression*: probit link

Binary data and aggregated ($N > 1$ data) are handled slightly differently.




```
library(ggplot2)
theme_set(theme_bw())
library(grid)
zmargin <- theme_update(panel.spacing=unit(0,"lines"))
library(dotwhisker)
library(descr)  ## for R^2 measures
library(aods3)  ## for overdispersion
library(arm)    ## binnedplot
library(dplyr)  ## tidyverse!
library(DescTools)
```

## *Contraception data example*

```
data("Contraception",package="mlmRev")
head(Contraception)
```

```
##   woman district use livch      age urban
## 1     1        1   N    3+  18.4400     Y
## 2     2        1   N     0  -5.5599     Y
## 3     3        1   N     2   1.4400     Y
## 4     4        1   N    3+   8.4400     Y
## 5     5        1   N     0 -13.5590     Y
## 6     6        1   N     0 -11.5600     Y
```

See here for more documentation.
**Given these variables, what model do we think we want to use?**
Visualize! Try some ggplots (univariate graphs are OK but multivariate graphs are almost always more informative . . . )

```
gg0 <- ggplot(Contraception,aes(age,use,colour=urban))+
    stat_sum(alpha=0.5)+facet_wrap(~livch,labeller=label_both)
gg0 + geom_smooth(aes(group=1))
```

Hard to summarize 0/1 values!
Alternative approach: binning (also see Faraway). (Transform!)

```
## transform via tidyverse ...
cc <- (Contraception
    %>% mutate(
            ## numeric (0/1) version of 'uses contraception'
            use_n=as.numeric(use)-1)
)
cc_agg0 <- (cc
    %>% group_by(livch,urban,age)
    %>% summarise(prop=mean(use_n),
            n=length(use),
            se=sqrt(prop*(1-prop)/n))
)
```

Plot:

```
ggplot(cc_agg0,aes(age,prop,colour=urban))+
    geom_pointrange(aes(ymin=prop-2*se,
                        ymax=prop+2*se))+
    facet_wrap(~livch,labeller=label_both)
```

Bin more coarsely:

```
## specify categories; compute midpoints as well
age_breaks <- seq(-15,20,by=5)
age_mids <- (age_breaks[-1]+age_breaks[-length(age_breaks)])/2
```

```
            ## discrete age categories
            age_cat=cut(age,breaks=age_breaks))
)
cc_agg <- (cc
    %>% mutate(age_cat=cut(age,breaks=age_breaks))
    %>% group_by(age_cat,urban,livch)
    %>% summarise(
            prop=mean(use_n),
            n=length(use),
            se=sqrt(prop*(1-prop)/n)
        )
    ## numeric values of age categories
    %>% mutate(age_mid=age_mids[as.numeric(age_cat)])
)
```

```
## Error:  <text>:5:47:  unexpected ')'
## 4:            ## discrete age categories
## 5:            age_cat=cut(age,breaks=age_breaks))
##                                                 ^
```

Plot:

```
## use numeric response rather than Y/N response
gg0B <- ggplot(cc,aes(age,use_n,colour=urban))+
    stat_sum(alpha=0.5)+facet_wrap(~livch,labeller=label_both)
gg_bin <- gg0B+geom_pointrange(data=cc_agg,
                    aes(x=age_mid,
                        y=prop,
                        ymin=prop-2*se,
                        ymax=prop+2*se,
                        size=n),
                    alpha=0.5)+
    scale_size(range=c(0.5,2))
```

```
## Error in fortify(data):  object 'cc_agg' not found
```

**How should we adjust our model specification based on this information?**

---

Suppose we use a model with a quadratic function of age plus all three-way interactions:

```
model1 <- glm(use_n ~ urban*(age+I(age^2))*livch,
              data=cc,
              family=binomial,
```

```
              x=TRUE  ## include model matrix in output
              )
```

Explore diagnostics (`plot()`; `DHARMa::simulateResiduals()`; `arm::binnedplot`; `mgcv::qq.gam`).

**Quantile residuals** [1] overcome many of the problems of GLM diagnostics, at the price of lots more computation.

[1] Ben, M. G. and V. J. Yohai (2004, March). Quantile-Quantile Plot for Deviance Residuals in the Generalized Linear Model. *Journal of Computational and Graphical Statistics 13*(1), 36–47; and Hartig, F. (2018). *DHARMa: Residual Diagnostics for Hierarchical (Multi-Level / Mixed) Regression Models*. R package version 0.2.0

```
plot(model1)  ## ugh!
arm::binnedplot(fitted(model1),residuals(model1))
DHARMa::simulateResiduals(model1,plot=TRUE)
mgcv::qq.gam(model1,pch=1)
```

If you really need a global goodness-of-fit test: **Hosmer-Lemeshow test** (very common) dominated by Cessie-van Houwelingen test [2].

[2] le Cessie, S. and J. C. van Houwelingen (1991, December). A goodness-of-fit test for binary regression models, based on smoothing methods. *Biometrics 47*(4), 1267–1282; and Hosmer, D. W., T. Hosmer, S. L. Cessie, and S. Lemeshow (1997, May). A Comparison of Goodness-of-Fit Tests for the Logistic Regression Model. *Statistics in Medicine 16*(9), 965–980

```
DescTools::HosmerLemeshowTest(fit=fitted(model1),
                              obs=model1$y,
                              X=model1$x)
```

### *pseudo-$R^2$ measures*

The UCLA statistics site has a very nice description of pseudo-$R^2$ measures.

- fraction of variance explained
- model improvement
- fraction of deviance explained: (dev(null)-dev(model))/dev(null) ("McFadden"):

```
with(model3,1-deviance/null.deviance)

## Error in with(model3, 1 - deviance/null.deviance):
## object 'model3' not found
```

- correlation ("Efron"):

```
cor(cc$use_n,predict(model3,type="response"))^2

## Error in predict(model3, type = "response"):  object
## 'model3' not found
```

- Cox and Snell: average deviance explained

$$1 - (L(\text{null})/L(\text{full}))^{2/n}$$

  (i.e. look at proportion on the likelihood scale, not the log-likelihood scale)

- Nagelkerke: Cox and Snell, adjusted to max=1

```
descr::LogRegR2(model3)
```

```
## Error in descr::LogRegR2(model3):  object 'model3' not
found
```

*Plot predictions*

```
gg_bin+geom_smooth(method="glm",
                   method.args=list(family=binomial),
                   formula=y~x+I(x^2)
                   )
```

```
## Error in eval(expr, envir, enclos):  object 'gg_bin' not
found
```

Or by hand: `predict` function.
Confidence intervals: get new model matrix and compute $XVX^T$ to get variances on the link-function scale. Then compute Normal CIs on the link scale, *then* back-transform. Or use `se=TRUE` in `predict`.

```
pvar <- newX %*% vcov(g1) %*% t(newX)
```

```
## Error in eval(expr, envir, enclos):  object 'newX' not
found
```

```
pse <- sqrt(diag(pvar))
```

```
## Error in diag(pvar):  object 'pvar' not found
```

Or equivalently for any model type where `predict` has an `se.fit` argument:

```
pse <- predict(g1,newdata=newdata,se.fit=TRUE)$se.fit
```

```
## Error in predict(g1, newdata = newdata, se.fit = TRUE):
object 'g1' not found
```

```
lwr0 <- pred0-2*pse  ## or qnorm(0.025)
```

```
## Error in eval(expr, envir, enclos):  object 'pred0' not
found

upr0 <- pred0+2*pse  ## or qnorm(0.975)

## Error in eval(expr, envir, enclos):  object 'pred0' not
found

lwr <- plogis(lwr0)

## Error in plogis(lwr0):  object 'lwr0' not found

upr <- plogis(upr0)

## Error in plogis(upr0):  object 'upr0' not found
```

Note:

- back-transforming the standard errors via a logistic usually doesn't make sense: if you want to back-transform them (approximately), you have to multiply them by $(d\mu/d\eta)$, i.e. use `dlogis`.
- if you use `response=TRUE` and `se.fit=TRUE`, R computes the standard errors, scales them as above, and uses them to compute (approximate) *symmetric* confidence intervals. Unless your sample is very large and/or your predicted probabilities are near 0.5 (so the CIs don't get near 0 or 1), it's probably best to use the approach above

```
## prediction frame: all combinations of variables
pframe <- with(Contraception,
               expand.grid(age=unique(age),
                           livch=levels(livch),
                           urban=levels(urban)))
predfun <- function(model) {
    pp <- predict(model, newdata=pframe, type="link", se.fit=TRUE)
    linkinv <- family(model)$linkinv
    pframe$use_n <- linkinv(pp$fit)
    pframe$lwr <- linkinv(pp$fit-2*pp$se.fit)
    pframe$upr <- linkinv(pp$fit+2*pp$se.fit)
    return(pframe)
}
pp1 <- predfun(model1)
```

### *Posterior predictive simulations*

Pick a summary statistic that matters (e.g.

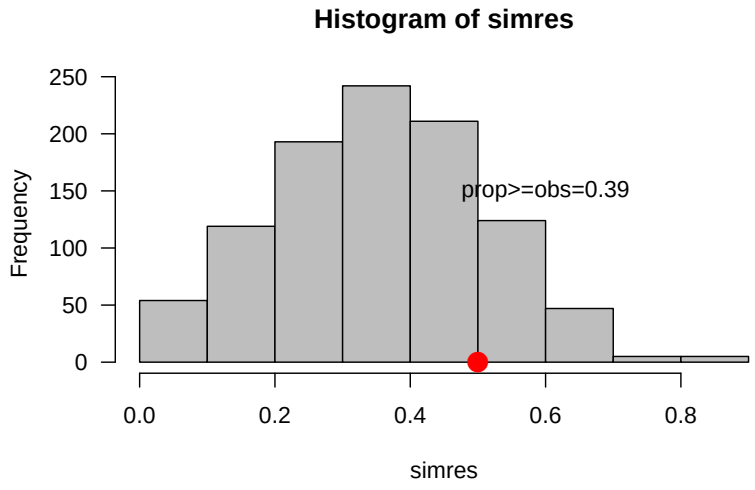
```
ppfun <- function(dd) {
    w <- which(dd$urban=="Y" & dd$livch=="0" & abs(dd$age)<1)
    return(mean(dd$use_n[w]))
}
ppfun(cc)

## [1] 0.5

ss <- simulate(model1,1000)
simres <- rep(NA,1000)
newcc <- cc
for (i in 1:1000) {
    newcc$use_n <- ss[,i]
    simres[i] <- ppfun(newcc)
}
```

Plot results:

```
par(las=1)
hist(simres,col="gray")
points(ppfun(cc),0,col="red",cex=2,pch=16)
p_upr <- mean(simres>=ppfun(cc))
p_lwr <- mean(simres<=ppfun(cc))
text(0.6,150,paste0("prop>=obs=",round(p_upr,2)))
```




```
## 2-tailed p-value
2*min(p_upr,p_lwr)

## [1] 0.784
```

*Simplify model*

With caution!

```
drop1(model1,test="Chisq")

## Single term deletions
##
## Model:
## use_n ~ urban * (age + I(age^2)) * livch
##                      Df Deviance    AIC     LRT Pr(>Chi)
## <none>                    2400.9 2448.9
## urban:age:livch       3   2401.2 2443.2 0.26485   0.9665
## urban:I(age^2):livch  3   2401.8 2443.8 0.89356   0.8270

## three-way interactions NS?
model2 <- update(model1, . ~ (urban+(age+I(age^2)+livch))^2)
drop1(model2,test="Chisq")

## Single term deletions
##
## Model:
## use_n ~ urban + age + I(age^2) + livch + urban:age + urban:I(age^2) +
##     urban:livch + age:I(age^2) + age:livch + I(age^2):livch
##                Df Deviance    AIC    LRT Pr(>Chi)
## <none>              2402.1 2440.1
## urban:age       1   2402.1 2438.1 0.0068  0.93452
## urban:I(age^2)  1   2403.0 2439.0 0.9059  0.34120
## urban:livch     3   2404.4 2436.4 2.2447  0.52320
## age:I(age^2)    1   2402.1 2438.1 0.0005  0.98302
## age:livch       3   2409.8 2441.8 7.6792  0.05313 .
## I(age^2):livch  3   2403.4 2435.4 1.3214  0.72405
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

## two-way interactions NS?
model3 <- update(model1, . ~ (urban+(age+I(age^2)+livch)))
## or LRT
anova(model1,model2,model3,test="Chisq")

## Analysis of Deviance Table
##
## Model 1: use_n ~ urban * (age + I(age^2)) * livch
## Model 2: use_n ~ urban + age + I(age^2) + livch + urban:age + urban:I(age^2) +
##     urban:livch + age:I(age^2) + age:livch + I(age^2):livch
## Model 3: use_n ~ urban + age + I(age^2) + livch
```

```
##   Resid. Df Resid. Dev  Df Deviance Pr(>Chi)
## 1      1910     2400.9
## 2      1915     2402.1  -5  -1.2276   0.9422
## 3      1927     2417.7 -12 -15.5305   0.2137
```

```
car::Anova(model3)

## Analysis of Deviance Table (Type II tests)
##
## Response: use_n
##          LR Chisq Df Pr(>Chisq)
## urban      52.849  1  3.602e-13 ***
## age         0.265  1      0.607
## I(age^2)   39.070  1  4.088e-10 ***
## livch      33.333  3  2.739e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

drop1(model3,test="Chisq")

## Single term deletions
##
## Model:
## use_n ~ urban + age + I(age^2) + livch
##          Df Deviance    AIC    LRT  Pr(>Chi)
## <none>        2417.7 2431.7
## urban     1   2470.5 2482.5 52.849 3.602e-13 ***
## age       1   2417.9 2429.9  0.265     0.607
## I(age^2)  1   2456.7 2468.7 39.070 4.088e-10 ***
## livch     3   2451.0 2459.0 33.333 2.739e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
dw1 <- dwplot(model3)+geom_vline(xintercept=0,lty=2)
dw2 <- dwplot(model3,by_2sd=FALSE)+geom_vline(xintercept=0,lty=2)
```

Can compare the effect of dropping interactions (carefully!)

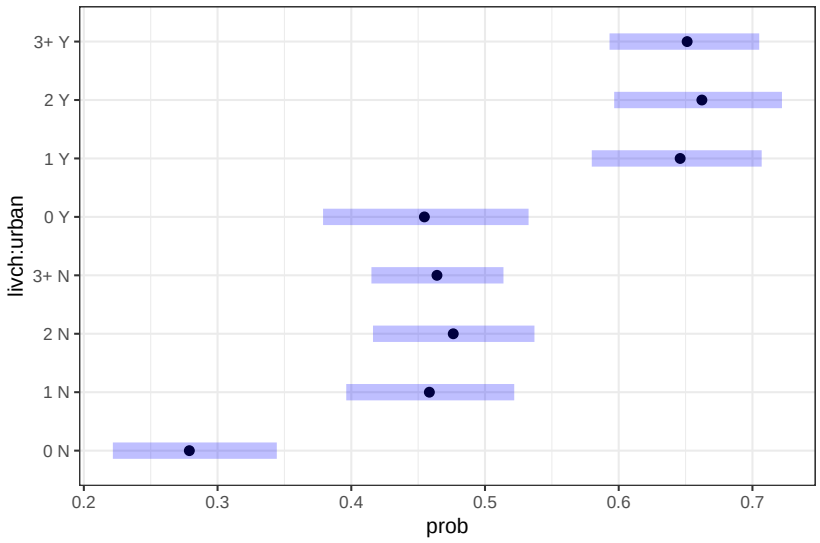
```
mod_list <- list(full=model1,twoway=model2,reduced=model3)
dw_comb <- dwplot(mod_list)+ geom_vline(xintercept=0,lty=2)
```

```
pp_list <- lapply(mod_list,predfun)
pp_frame <- dplyr::bind_rows(pp_list,.id="method")
gg_compare_pred <- gg0 + geom_line(data=pp_frame,
                                   aes(linetype=method))
```

```
summary(model3)
```

```
##
## Call:
## glm(formula = use_n ~ urban + age + I(age^2) + livch, family = binomial,
##     data = cc, x = TRUE)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4738  -1.0369  -0.6683   1.2401   1.9765
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.9499521  0.1560118  -6.089 1.14e-09 ***
## urbanY       0.7680975  0.1061916   7.233 4.72e-13 ***
## age          0.0045837  0.0089084   0.515    0.607
## I(age^2)    -0.0042865  0.0007002  -6.122 9.23e-10 ***
## livch1       0.7831128  0.1569096   4.991 6.01e-07 ***
## livch2       0.8549040  0.1783573   4.793 1.64e-06 ***
## livch3+      0.8060251  0.1784817   4.516 6.30e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2590.9  on 1933  degrees of freedom
## Residual deviance: 2417.7  on 1927  degrees of freedom
## AIC: 2431.7
##
## Number of Fisher Scoring iterations: 4
```

```
plot(emmeans::emmeans(model3,~livch*urban,type="response"))
```




*Confidence intervals on predictions etc.*

(delta method; bootstrap; simulation)

## *References*

Ben, M. G. and V. J. Yohai (2004, March). Quantile-Quantile Plot for Deviance Residuals in the Generalized Linear Model. *Journal of Computational and Graphical Statistics 13*(1), 36–47.

Hartig, F. (2018). *DHARMa: Residual Diagnostics for Hierarchical (Multi-Level / Mixed) Regression Models*. R package version 0.2.0.

Hosmer, D. W., T. Hosmer, S. L. Cessie, and S. Lemeshow (1997, May). A Comparison of Goodness-of-Fit Tests for the Logistic Regression Model. *Statistics in Medicine 16*(9), 965–980.

le Cessie, S. and J. C. van Houwelingen (1991, December). A goodness-of-fit test for binary regression models, based on smoothing methods. *Biometrics 47*(4), 1267–1282.

McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models*. London: Chapman and Hall.