Tidy models

Ben Bolker

October 25, 2018

Introduction

The `broom` package is the core of the tidymodels ecosystem, a set of tools for handling models in a "tidy" fashion. (`broom.mixed` contains extensions for mixed models, and a few other odds and ends.)

- `tidy` produces coefficient tables (estimates, standard errors, t/Z statistics, p values, confidence intervals, etc.)

  - use `conf.int=TRUE` to get confidence intervals (not just SEs)
  - defaults to Wald CIs; use `conf.method="profile"` for profile CIs
  - use `exponentiate=TRUE` to back-transform log to response scale, logit to odds scale, ...

- `glance` produces model summaries (R^2, residual df, AIC, BIC, etc.)

- `augment` produces predictions, residuals, etc..

Revisiting the AIDS data

```
library(glmmTMB)
library(dotwhisker)
library(ggplot2)
library(broom)
library(broom.mixed)
aids <- read.csv("../data/aids.csv")
aids <- transform(aids,
                  date=year+(quarter-1)/4,
                  index=seq(nrow(aids)))
```

```
g1 <- glm(cases ~ poly(date,2), data=aids, family=poisson)
g2 <- update(g1, family=quasipoisson)
g3 <- glmmTMB::glmmTMB(cases~poly(date,2), data=aids, family=nbinom2)
```

```
## Warning in fitTMB(TMBStruc):  Model convergence problem;
non-positive-definite Hessian matrix.  See vignette('troubleshooting')
## Warning in fitTMB(TMBStruc):  Model convergence problem;
false convergence (8).  See vignette('troubleshooting')

g4 <- glmmTMB::glmmTMB(cases~poly(date,2), data=aids, family=nbinom1)
```
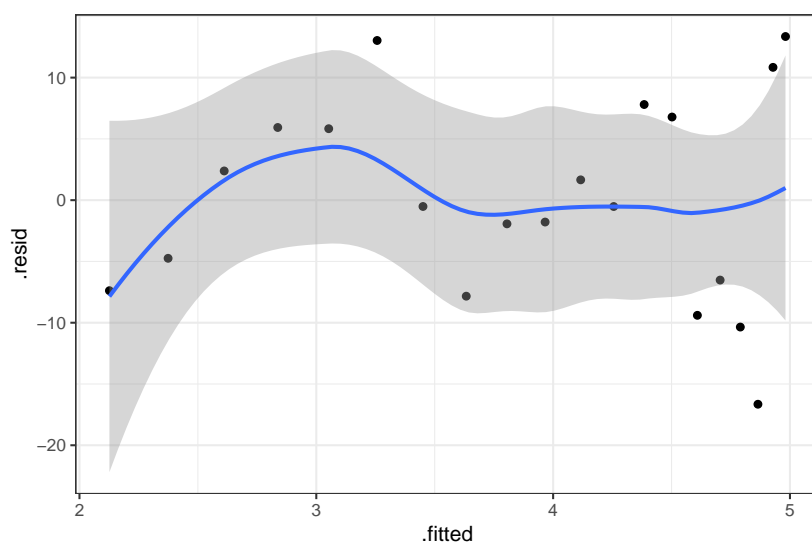
```
aa <- augment(g4,data=aids)
```

```
## fitted-residual plot
ggplot(aa,aes(.fitted,.resid))+geom_point()+geom_smooth()

## 'geom_smooth()' using method = 'loess' and formula 'y ~
x'
```
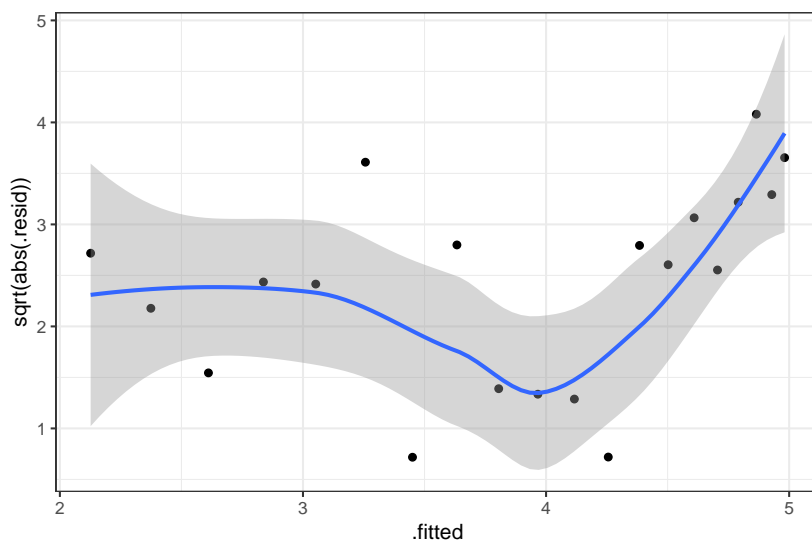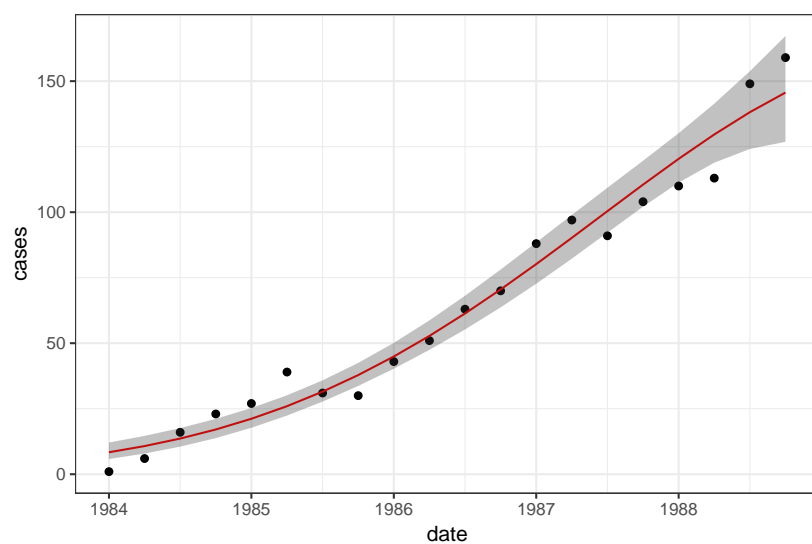


```
## scale-location plot
ggplot(aa,aes(.fitted,sqrt(abs(.resid))))+geom_point()+geom_smooth()

## 'geom_smooth()' using method = 'loess' and formula 'y ~
x'
```

```
## manual addition of predictions to the plot
ggplot(aa,aes(date,cases))+geom_point()+
    geom_line(aes(y=exp(.fitted)),colour="red")+
    geom_ribbon(aes(ymin=exp(.fitted-1.96*.se.fit),
                    ymax=exp(.fitted+1.96*.se.fit)),
                alpha=0.3)
```


dotwhisker::dwplot

dotwhisker::dwplot creates *coefficient plots*

- works under the hood by calling tidy for one or more models and combining the results

- can pass arguments to `tidy` (e.g. `conf.method`)

- by default, scales continuous predictors by 2σ; use `by_2sd=FALSE` to turn this off

- drops intercept by default

- given a (named) list of models, plots all the coefficients side-by-side (use to compare different modeling approaches, or models with different subsets of predictors)

- creates a `ggplot` object that can be post-processed by adding components, faceting, changing colour scales, etc..

  Example:

```
dwplot(list(poisson=g1,QP=g2,nb2=g3,nb1=g4),
       by_2sd=FALSE)+
   geom_vline(xintercept=0,lty=2)
```

```
## Warning in sqrt(diag(object$cov.fixed)):  NaNs produced
```
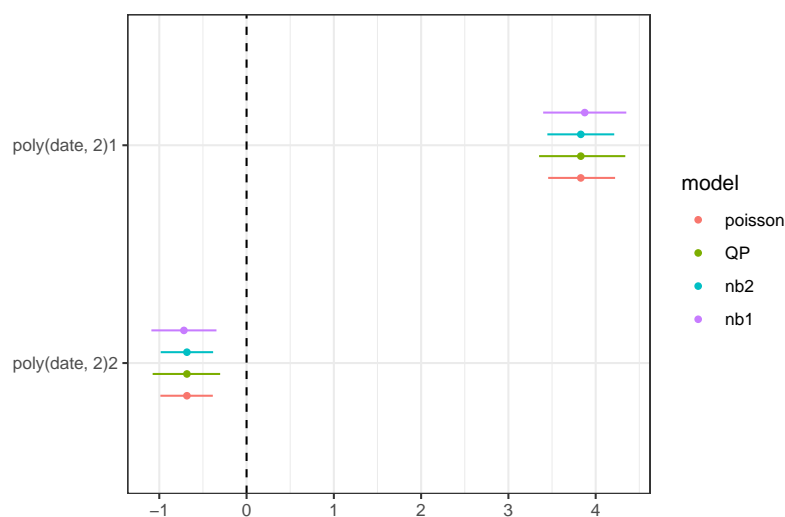

huxtable::huxreg

If your boss insists on parameter tables:

```
library(huxtable)
h0 <- huxreg(poisson=g1,QP=g2,nbinom2=g3,nbinom1=g4)
```

```
## Warning in huxreg(poisson = g1, QP = g2, nbinom2 = g3,
nbinom1 = g4):  Unrecognized statistics:  r.squared
```

```
  ## Try setting "statistics" explicitly in the call to
huxreg()

cat(to_latex(h0,tabular_only=TRUE))
```

	poisson	QP	nbinom2	nbinom1
(Intercept)	3.869 ***	3.869 ***	3.869 ***	3.863 ***
	(0.039)	(0.050)	(0.039)	(0.048)
poly(date, 2)1	3.829 ***	3.829 ***	3.829 ***	3.875 ***
	(0.195)	(0.252)	(0.195)	(0.243)
poly(date, 2)2	-0.683 ***	-0.683 **	-0.683 ***	-0.718 ***
	(0.153)	(0.197)	(0.153)	(0.190)
N	20	20	20	20
logLik	-72.145			-71.277
AIC	150.290			150.555

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$.