Generalized Linear Models

We have seen linear regression and logistic regression, both of which are examples of generalized linear models (GLMs), but there are many other possible generalized linear models.

A generalized linear model can also include other parameters such as variance or overdispersion terms and/or cutpoints for latent responses.

When formally writing out a GLM, it is important to include these three components:

Chapter 15 in ROS mentions another set of GLMs which we will cover:

- Poisson and negative binomial models for count data
- logistic-binomial model, where y_i is a set of n_i Bernoulli trials and a related (beta-binomial model)
- The probit model for binary data
- Multinomial logit (and probit models) for categorical data, both ordered and unordered
- Robust regression, using non-normal errors for continuous data.

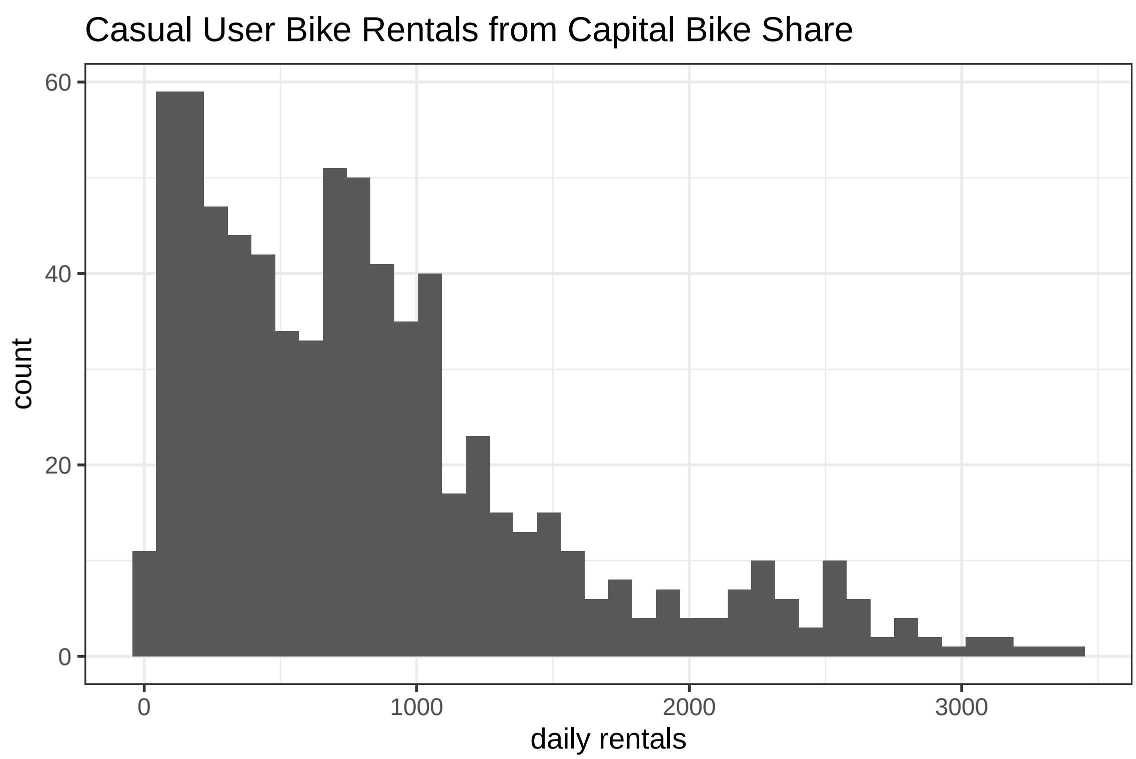
Count Regression

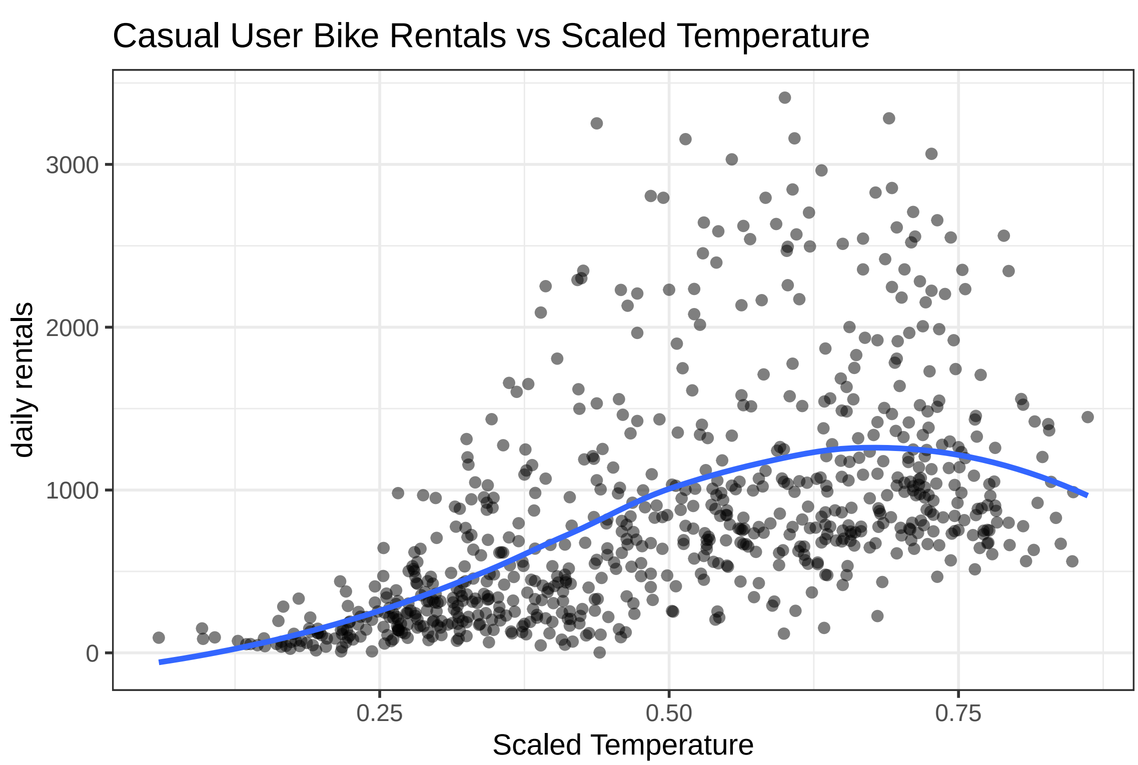
The motivating example that we will use for this scenario is a dataset that contains the total number of daily bike rentals from the Capital Bikeshare System in Washington, DC.

```
bikes <- read_csv("https://raw.githubusercontent.com/STAT506/GLM_Lectures/main/daily_bike.csv")
```

```
## Parsed with column specification:
## cols(
##   instant = col_double(),
##   dteday = col_character(),
##   season = col_double(),
##   yr = col_double(),
##   mnth = col_double(),
##   holiday = col_double(),
##   weekday = col_double(),
##   workingday = col_double(),
##   weathersit = col_double(),
##   temp = col_double(),
##   atemp = col_double(),
##   hum = col_double(),
##   windspeed = col_double(),
##   casual = col_double(),
##   registered = col_double(),
##   cnt = col_double()
## )
```

```
bikes <- bikes %>% mutate(temp_centered = scale(temp))
```


Casual User Bike Rentals from Capital Bike Share
count
60
40
20
0
0
1000
2000
3000
daily rentals


Casual User Bike Rentals vs Scaled Temperature
daily rentals
3000
2000
1000
0
0.25
0.50
0.75
Scaled Temperature

In the presence of overdispersion, negative binomial regression is a common solution.

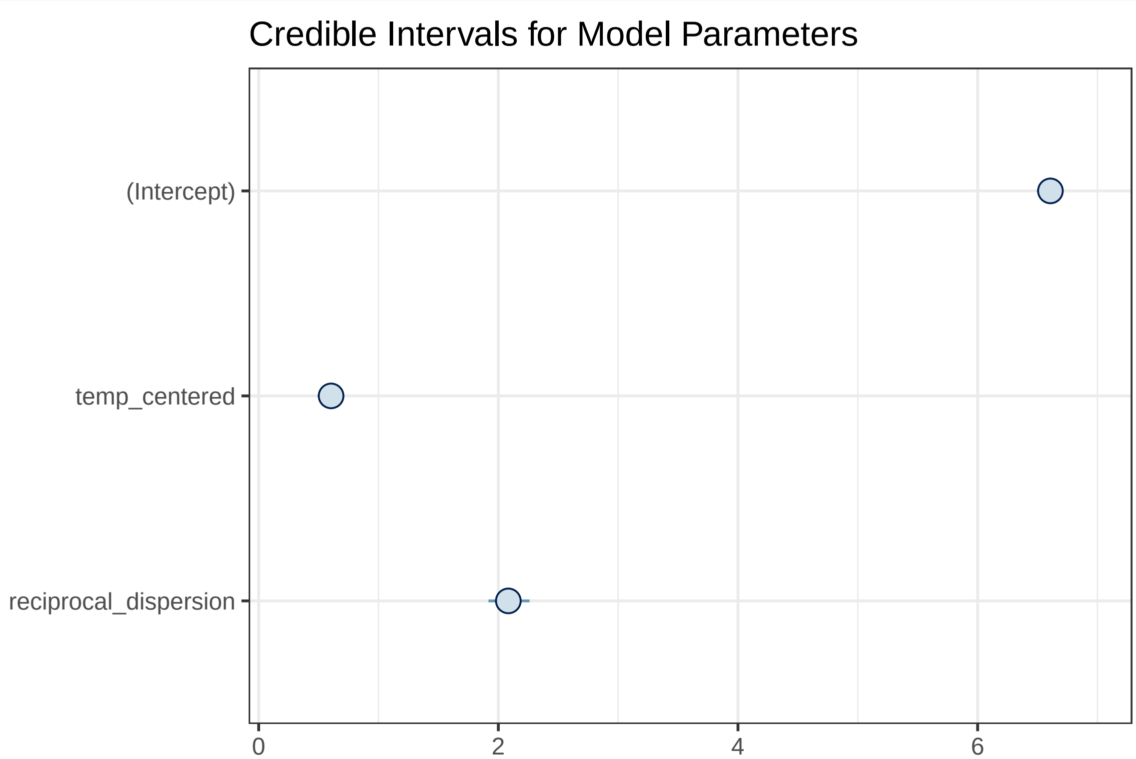
The parameter ϕ can account for additional dispersion in the model. Formally, the standard deviation of $y|x = \sqrt{E(y|x) + E(y|x)/\phi}$. So when $\phi \to \infty$ this limit results in a Poisson distribution.

Model Fitting and Intepreting Coefficients

```r
nb_model <- stan_glm(casual ~ temp_centered, family = neg_binomial_2, data = bikes, refresh = 0)
print(nb_model)
```

```
## stan_glm
##  family:       neg_binomial_2 [log]
##  formula:      casual ~ temp_centered
##  observations: 731
##  predictors:   2
## ------
##               Median MAD_SD
## (Intercept)   6.6    0.0
## temp_centered 0.6    0.0
##
## Auxiliary parameter(s):
##                       Median MAD_SD
## reciprocal_dispersion 2.1    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```r
plot(nb_model) + theme_bw() + ggtitle('Credible Intervals for Model Parameters')
```



So in the case $\exp(\beta_0) = 741$ with a credible interval of (705, 778).

Given that the model can be simplified as $y \sim NB(\exp(\beta_0 + \beta_1 x), \phi)$, the coefficient can also be interpreted in a multiplicative way. In particular $\exp(\beta_1)$ is the multiplicative change in y for a one unit change in x.

So with this model $\exp(\beta_1) = 1.83$ with a credible interval of (1.73, 1.94).

Count GLMS with Exposure

In other situations, the exposure could vary and the model could explicitly include this in the model.

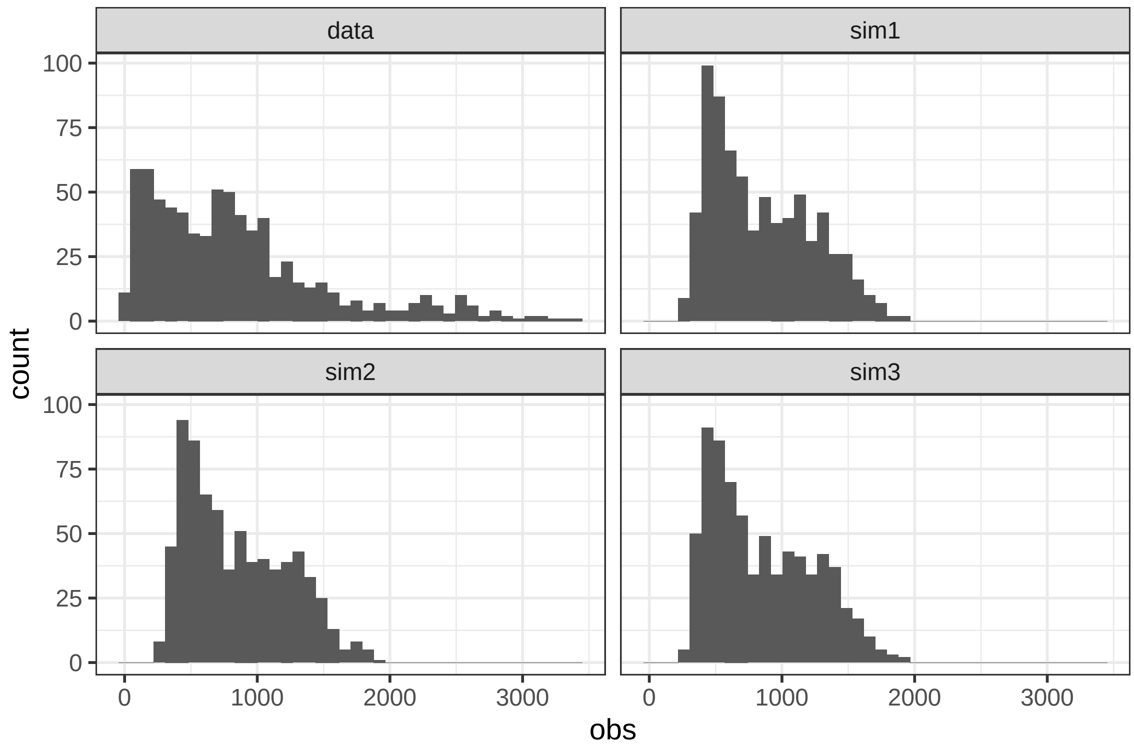
$$y_i \sim NB(u_i \theta_i, \psi),$$

As with other regression frameworks, posterior predictive checks can be a useful tool for model checking.

```
pois_model <- stan_glm(casual ~ temp_centered, family = poisson, data = bikes, refresh = 0)
pp <- posterior_predict(pois_model)
```

This can either be done visually

```
tibble(obs = c(bikes$casual, pp[1,], pp[2,], pp[3,]),
       group = rep(c('data','sim1','sim2','sim3'), each = ncol(pp))) %>%
  ggplot(aes(x = obs)) + geom_histogram(bins = 40) + facet_wrap(.~ group) + theme_bw()
```



or by comparing summary statistics between the simulated datasets and the observed data.

```
sim_max = apply(pp,1,max)
data_max = max(bikes$casual)
tibble(sim_max = sim_max) %>% ggplot(aes(x = sim_max)) + geom_histogram(bins = 50 ) +
  geom_vline(xintercept = data_max) + theme_bw() +
  ggtitle("Comparison of maximum value from simulation vs model fit")
```

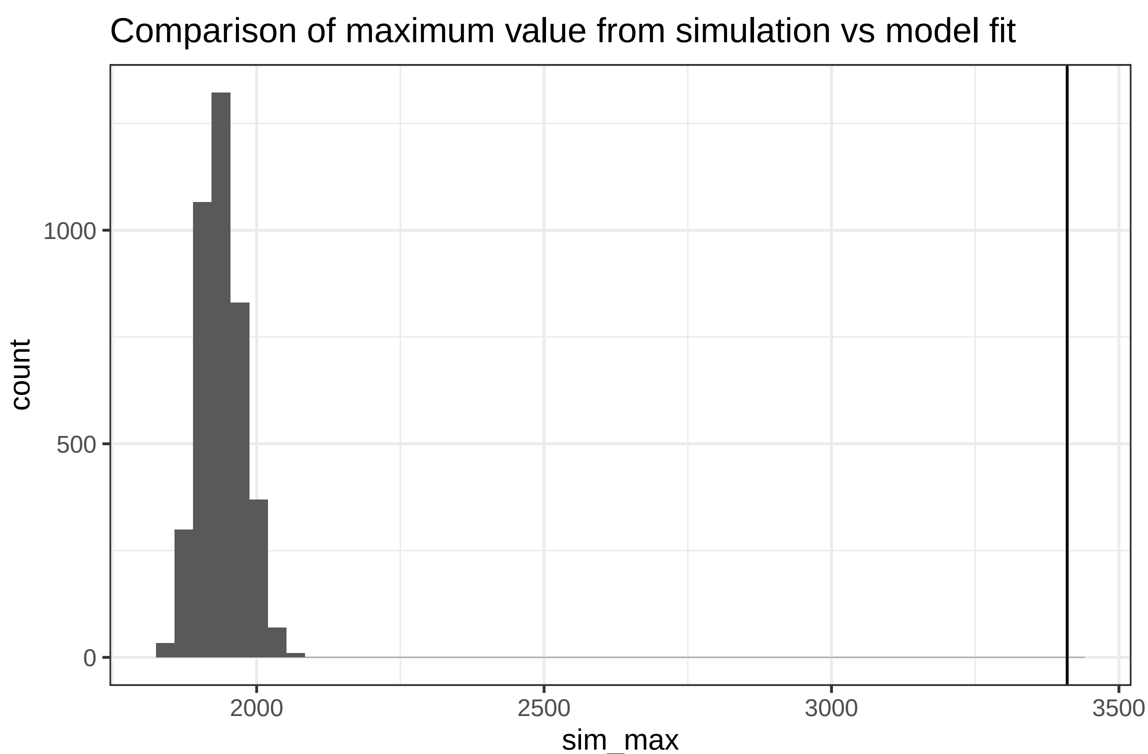


Figure 1: Vertical line represents maximum value in observed dataset