

Generalized Linear Models, part 1

Frank Edwards

3/27/2020

Generalized linear models

The components of a model

- Likelihood of the data (the small-world data generator): *e.g.*
 $y \sim \text{Binomial}(n, p)$
- The linear function and link function (converts likelihood parameters to linear combinations of predictors): *e.g.* $\text{logit}(p) = \alpha + \beta x$
- Priors (our initial beliefs about plausible parameter values) *e.g.*:

$$\alpha \sim \text{Normal}(0, 1)$$

$$\beta \sim \text{Normal}(0, 1)$$

The Binomial count model

When $n = 1$, a Binomial regression model with a logit link function is called a logistic regression. When $n > 1$, we are instead modeling counts of events as our outcome.

The expected value for a Binomial variable with probability p and number of trials n is np , and the variance is $np(1 - p)$.

The Binomial count model

When $n = 1$, a Binomial regression model with a logit link function is called a logistic regression. When $n > 1$, we are instead modeling counts of events as our outcome.

The expected value for a Binomial variable with probability p and number of trials n is np , and the variance is $np(1 - p)$.

When n is large, and p is small, $np \approx np(1 - p)$. This special case of the Binomial distribution is the Poisson distribution.

The Poisson likelihood

Where y is a non-negative integer (count)

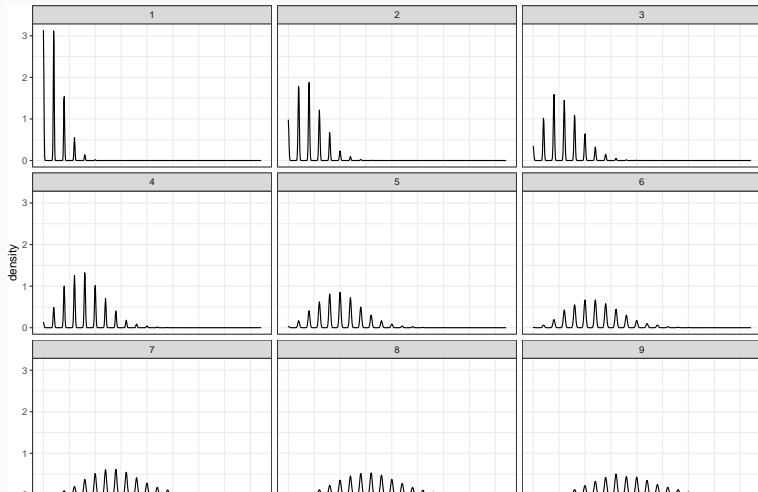
$$y \sim \text{Poisson}(\lambda)$$

$$E(y) = \bar{y} = \lambda$$

$$\text{Var}(y) = \lambda$$

The Poisson likelihood

```
ggplot(pois_demo, aes(x=count)) +  
  geom_density(adjust = 1/4) +  
  facet_wrap(~lambda)
```



Let's look at each Poisson variable

Let's look at each Poisson variable

```
pois_demo %>% group_by(lambda) %>%  
  summarise(mean = mean(count),  
            variance = var(count))
```

```
## # A tibble: 9 x 3  
##   lambda mean variance  
##   <int> <dbl>   <dbl>  
## 1     1  1.01    1.01  
## 2     2  2.00    1.99  
## 3     3  2.97    2.94  
## 4     4  4.00    3.99  
## 5     5  5.02    5.04  
## 6     6  6.02    6.27  
## 7     7  7.00    6.93  
## 8     8  7.98    8.11  
## 9     9  8.98    9.20
```

For a non-negative integer y

$$y_i \sim \text{Poisson}(\lambda)$$

$$\log \lambda_i = \alpha + \beta x$$

Using appropriate priors for α and β

The data for today: Fatal Encounters

Let's use data on police-involved killings to estimate how many men of different racial groups are killed by police in the average US county.

```
fe<-read_csv("./fe_demo.csv")  
### on your computer, run fe<-read_csv("./slides/fe_demo.csv")  
head(fe)
```

```
## # A tibble: 6 x 5  
##   fips state  pop deaths race.ethn  
##   <dbl> <chr> <dbl>   <dbl> <chr>  
## 1  1001 AL      483       0 latino  
## 2  1003 AL     3218       0 latino  
## 3  1005 AL      473       0 latino  
## 4  1007 AL      163       0 latino  
## 5  1009 AL     1713       0 latino  
## 6  1011 AL      359       0 latino
```

A mortality model using a Poisson likelihood

Let's define a Poisson regression of death rates d

$$d_i \sim \text{Poisson}(\lambda)$$

$$\log \lambda_i = \alpha[\text{race}]_i$$

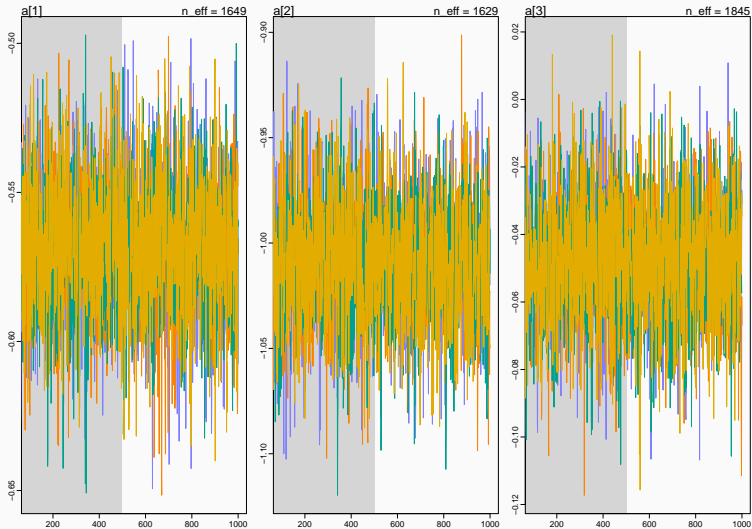
$$\alpha[\text{race}] \sim \text{Normal}(0, 2)$$

$$\sigma \sim \text{Exp}(1)$$

```
fe_model<-fe %>%  
  mutate(race_ethn = factor(race.ethn),  
         log_pop = log(pop + 1)) %>%  
  select(deaths, race_ethn, log_pop)  
  
fe_pois<-ulam(alist(  
  deaths ~ dpois(l),  
  log(l) <- a[race_ethn],  
  a[race_ethn] ~ dnorm(0,2)  
) , data = fe_model, chains = 4, cores = 2)
```

Check convergence

```
traceplot(fe_pois)
```



Log-scale parameters are hard to interpret...

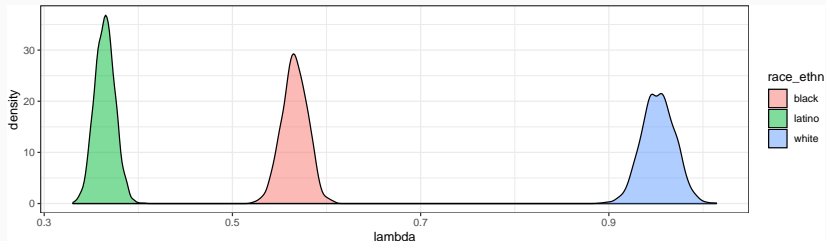
```
precis(fe_pois)
```

```
## [1] mean sd    5.5% 94.5% n_eff Rhat4  
## <0 rows> (or 0-length row.names)
```

Average expected deaths per county

```
post_a<-extract.samples(fe_pois)
post_a<-as.data.frame(post_a)
names(post_a)<-c("black", "latino", "white")
post_a<-post_a %>%
  pivot_longer(cols = everything(),
               names_to = "race_ethn",
               values_to = "log_lambda")
# inverse link function to get lambda on E(deaths) scale
post_a<-post_a %>%
  mutate(lambda = exp(log_lambda))

ggplot(post_a,
       aes(x = lambda, fill = race_ethn)) +
  geom_density(alpha = 0.5)
```



Problem: these don't adjust for population size

The standard Poisson model is *unbounded*, and assumes each observation is drawn from a similar unit. However, we can use Poisson models to estimate event *rates* using an *offset*.

Problem: these don't adjust for population size

The standard Poisson model is *unbounded*, and assumes each observation is drawn from a similar unit. However, we can use Poisson models to estimate event *rates* using an *offset*.

An offset (or exposure) scales λ to the size of the exposure, turning λ into a rate per unit of the offset.

Including an offset in the model

Let's now estimate a model that includes population p as an exposure variable (typically time, population, or some other measure of event opportunity).

$$d_i \sim \text{Poisson}(\lambda)$$

$$\log \frac{\lambda_i}{p_i} = \log \lambda_i - \log p_i = \alpha + \beta x_i$$

Including an offset in the model

Let's now estimate a model that includes population p as an exposure variable (typically time, population, or some other measure of event opportunity).

$$d_i \sim \text{Poisson}(\lambda)$$
$$\log \frac{\lambda_i}{p_i} = \log \lambda_i - \log p_i = \alpha + \beta x_i$$

Which we can rewrite as

$$\log \lambda_i = \log p_i + \alpha + \beta x_i$$

We add the offset to the regression without including a parameter.

Let's re-estimate the model with an offset

$$d_i \sim \text{Poisson}(\lambda)$$

$$\log \lambda_i = \log p_i + \alpha[\text{race}]_i$$

$$\alpha[\text{race}] \sim \text{Normal}(0, 2)$$

$$\sigma \sim \text{Exp}(1)$$

```
fe_pois_offset<-ulam(alist(  
  deaths ~ dpois(l),  
  log(l) <- log_pop + a[race_ethn],  
  a[race_ethn] ~ dnorm(0,2)  
) , data = fe_model, chains = 4, cores = 2)
```

Interpreting the model, checking convergence diagnostics

```
precis(fe_pois_offset, depth = 2)
```

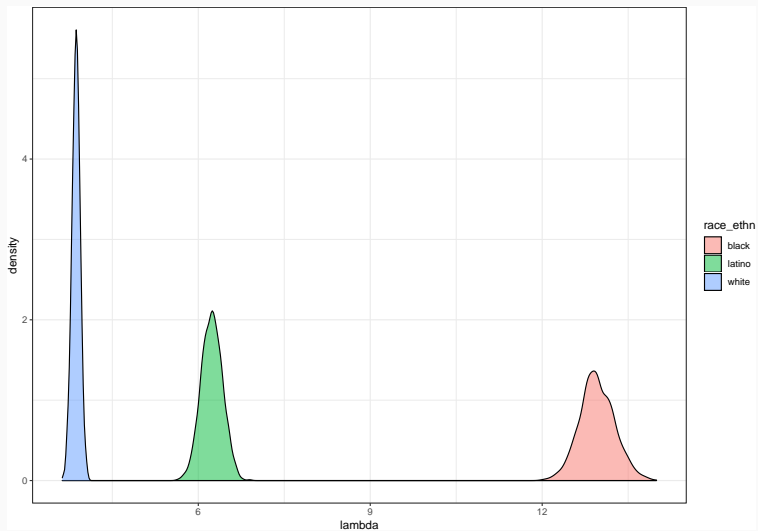
##	mean	sd	5.5%	94.5%	n_eff	Rhat4
## a[1]	-8.951482	0.02350278	-8.98850	-8.912533	1664.565	0.9984476
## a[2]	-9.682924	0.02969784	-9.73053	-9.635867	1854.218	1.0003190
## a[3]	-10.160687	0.01828711	-10.19149	-10.131874	1742.286	1.0013853

Let's check the expected number of deaths for each group assuming a population of 100,000.

```
sim_dat<-data.frame(race_ethn =  
                    c("black", "latino", "white"),  
                    log_pop = log(1e5))  
  
post_lambda<-link(fe_pois_offset, sim_dat)  
post_lambda<-as.data.frame(post_lambda)  
names(post_lambda)<-c("black", "latino", "white")  
  
post_lambda<-post_lambda %>%  
  pivot_longer(cols = everything(),  
               names_to = "race_ethn",  
               values_to = "lambda")
```

Posterior inference

```
ggplot(post_lambda,  
  aes(x = lambda, fill = race_ethn)) +  
  geom_density(alpha = 0.5)
```



Categorical regression

Categorical data falls into a fixed set of categories. It may be *unordered*, meaning that there is no inherent ranking of categories, or it may be *ordered*. Ordered categorical data has an explicit hierarchical ranking of values.

Are these variables ordered or unordered?

Are these variables ordered or unordered?

- Candidate choice in a primary election

Are these variables ordered or unordered?

- Candidate choice in a primary election
- Zip code for people choosing a place to move

Are these variables ordered or unordered?

- Candidate choice in a primary election
- Zip code for people choosing a place to move
- Cause of death

Are these variables ordered or unordered?

- Candidate choice in a primary election
- Zip code for people choosing a place to move
- Cause of death
- Opinions on a political issue on a thermometer / Likert scale
(e.g. Strongly oppose, oppose, neutral, support, strongly support)

Are these variables ordered or unordered?

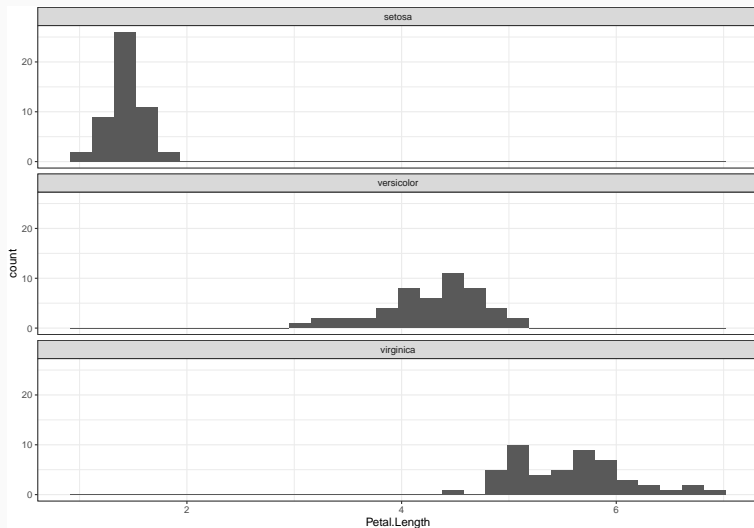
- Candidate choice in a primary election
- Zip code for people choosing a place to move
- Cause of death
- Opinions on a political issue on a thermometer / Likert scale
(e.g. Strongly oppose, oppose, neutral, support, strongly support)
- Graduate program to attend

Are these variables ordered or unordered?

- Candidate choice in a primary election
- Zip code for people choosing a place to move
- Cause of death
- Opinions on a political issue on a thermometer / Likert scale
(e.g. Strongly oppose, oppose, neutral, support, strongly support)
- Graduate program to attend
- Ranking of graduate program

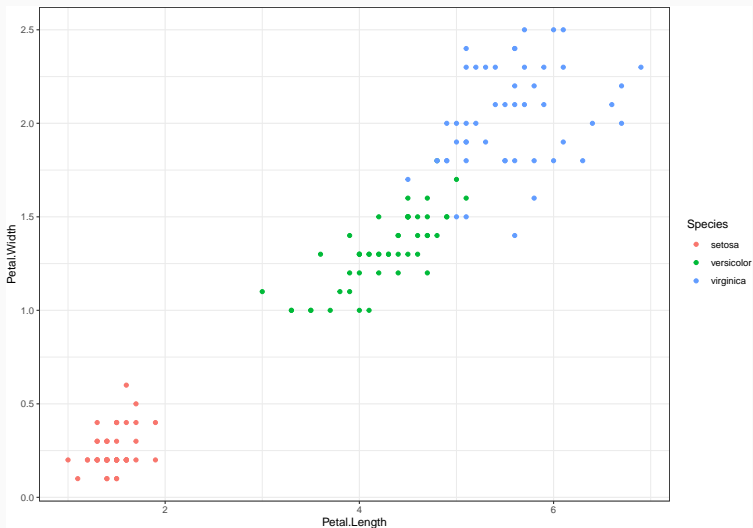
Visualizing categorical data, facets

```
ggplot(iris, aes(x = Petal.Length)) +  
  geom_histogram() +  
  facet_wrap(~Species, ncol=1)
```



Visualizing categorical data, color

```
ggplot(iris, aes(x = Petal.Length, y = Petal.Width,  
                 color = Species)) +  
  geom_point()
```



Predicting Iris species from petal length and width

Can we successfully classify Iris species based on the observed length of the petals?

We will use a *multinomial regression*, which is conceptually similar to running $k - 1$ logistic regressions, where k is the number of categories in our outcome.

Multinomial regression: basics

For a categorical outcome with K categories, estimate $K - 1$ models where 1,2,3 stand in for membership in group 1, 2, 3:

$$\log \frac{\Pr(y_i = 1)}{\Pr(y_i = K)} = \beta x_i$$

$$\log \frac{\Pr(y_i = 2)}{\Pr(y_i = K)} = \beta x_i$$

...

$$\log \frac{\Pr(y_i = K - 1)}{\Pr(y_i = K)} = \beta x_i$$

Key assumption: Independence of irrelevant alternatives. Odds of choice do not depend on the presence or absence of other alternatives (i.e. car vs bus or car vs red bus vs blue bus)

The book provides details on using Stan directly, but this is tricky with `ulam()`. I'm using the `brms` package here (I'll show you more of this later!)

```
library(brms)
```

```
iris_model<-brm(Species ~ Petal.Length,  
               family = categorical,  
               data = iris)
```

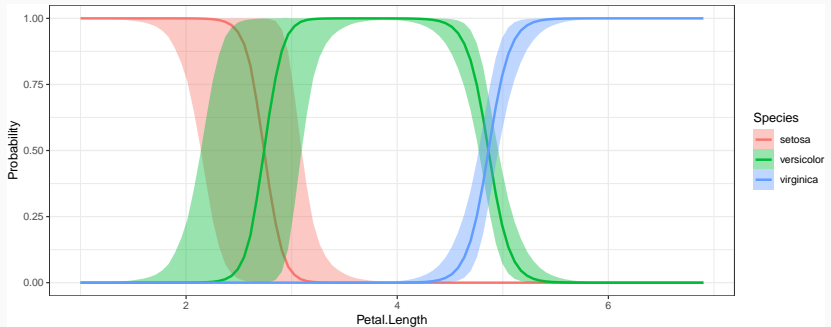
These are hard to interpret...

```
summary(iris_model)
```

```
## Family: categorical
## Links: muversicolor = logit; muvirginica = logit
## Formula: Species ~ Petal.Length
## Data: iris (Number of observations: 150)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## muversicolor_Intercept    -39.78    23.20  -101.41  -12.55  1.01     449
## muvirginica_Intercept     -89.08    27.29  -158.54  -51.70  1.01     494
## muversicolor_Petal.Length   14.54     7.87    4.91   35.28  1.02     412
## muvirginica_Petal.Length    24.68     8.53   13.25   46.58  1.01     432
##              Tail_ESS
## muversicolor_Intercept      429
## muvirginica_Intercept       474
## muversicolor_Petal.Length    401
## muvirginica_Petal.Length     407
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's predict flower species

```
conditional_effects(iris_model, categorical = T)
```



- We added two new likelihoods into our GLM tool-belt: the Poisson and the multinomial
- Poisson regression can handle counts and rates easily
- Multinomial regression models categorical outcomes
- Homework: Chapter 11 Easy and Medium questions. If you want practice with Poisson models, add 11H4. Note that `map == quap` and `map2stan == ulam`.