Week Nine

Last Week

- Exams

This Week: Generalized Linear Models for Count Data

Today:

- Exam Recap:
- Activity:
  - GLMs for count data
- Thursday: Lab
  - Separation

Next Week: Count Regression / Ordinal Regression

Probability Distributions for Count Data

Poisson Distribution:

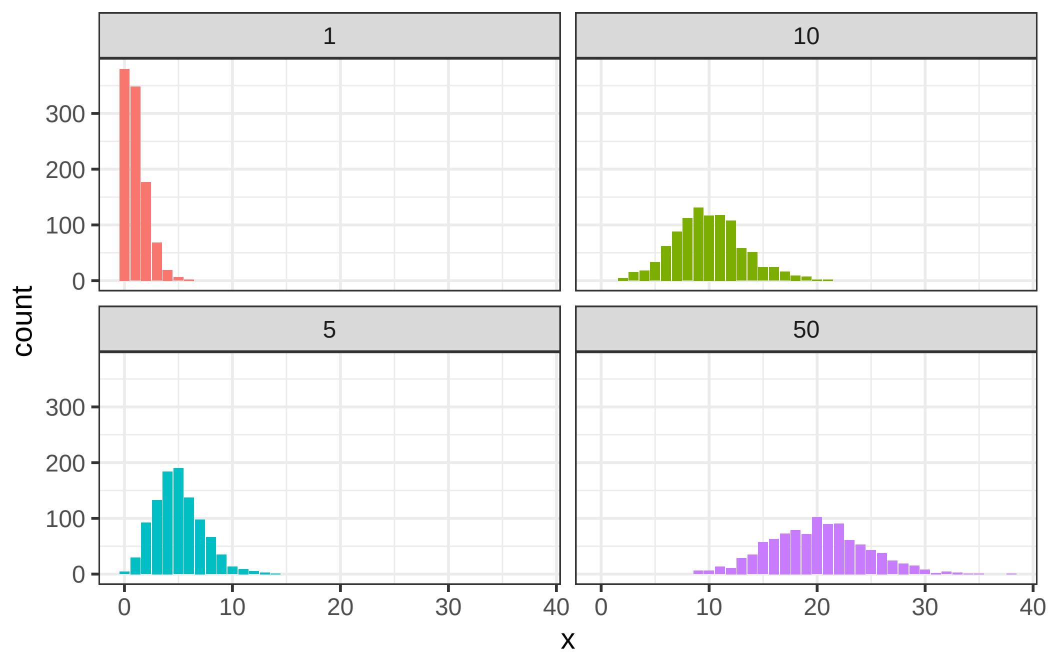
$$Pr[X = k] = \frac{\lambda^k \exp(-\lambda)}{k!}$$

- Expectation ($E[X] = \lambda$)
- Variance ($Var(X) = \lambda$)

Use `rpois()` to generate and visualize data with different λ parameters.

- $\lambda = [1, 5, 10, 20]$

```
library(tidyverse)
n <- 1000
tibble(x = c(rpois(n, 1), rpois(n, 5), rpois(n, 10), rpois(n,20)),
       lambda = rep(c('1', '5', '10', '50'), each = n)) |>
  ggplot(aes(x=x, fill = lambda)) + geom_bar() + theme_bw() +
  facet_wrap(.~lambda) + theme(legend.position = 'none')
```



Negative Binomial Distribution:

$$Pr[X = k] = \frac{\Gamma(k+n)}{\Gamma(n)k!} p^n (1-p)k$$

- Expectation ($E[X] = n(1-p)/p = \mu$)
- Variance ($Var(X) = n(1-p)/p^2$)

Alternatively, we can define

- the mean, $\mu = n(1-p)/p$
- the size (dispersion parameter), as $p = size/(size + mu)$ which implies $\rightarrow$ that the variance $= \mu + \mu^2/size$

Use **rnbinom()** with **mu** and **size** to simulate data with some different combinations of the parameters.

- $\mu = [1, 5, 10, 20]$
- size = [.75, 1, 10]

Then plot figures and confirm that the mean and variance of the data match your expectations.

```
n <- 5000

data20_75 <- rnbinom(n, mu = 20, size = .75)
mean(data20_75)
```

```
[1] 20.1474
```

```
var(data20_75)
```

```
[1] 567.8692
```

```
data20_1 <- rnbinom(n, mu = 20, size = 1)
mean(data20_1)
```

```
[1] 20.1662
```

```r
var(data20_1)
```

```
[1] 445.4201
```

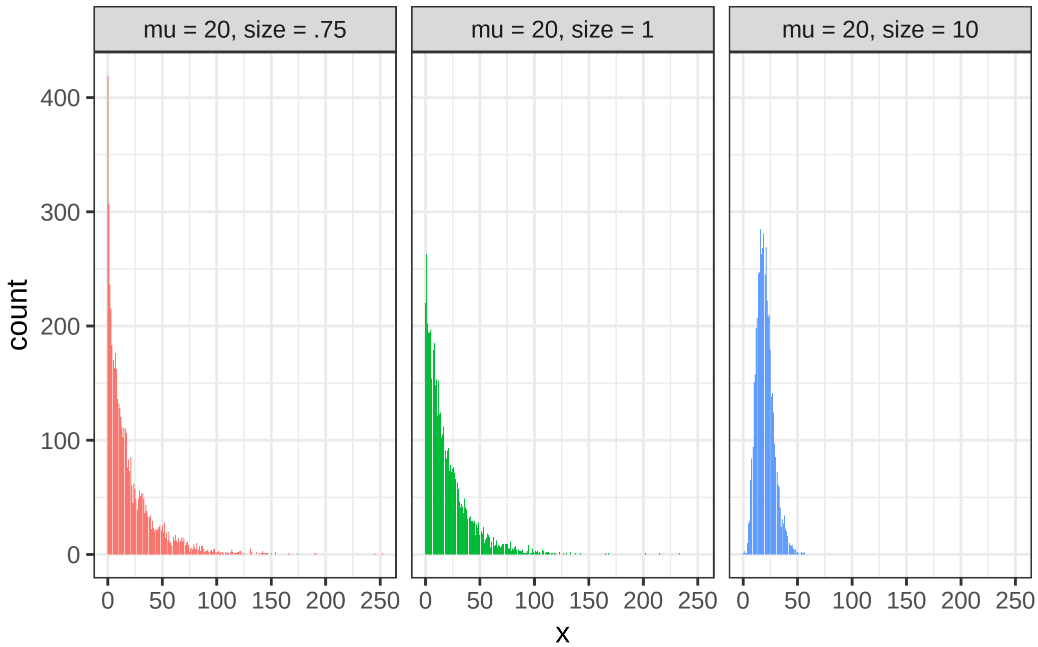
```r
data20_10 <- rnbinom(n, mu = 20, size = 10)
mean(data20_10)
```

```
[1] 19.921
```

```r
var(data20_10)
```

```
[1] 60.1804
```

```r
tibble(x = c(data20_75, data20_1, data20_10),
       params = rep(c('mu = 20, size = .75', 'mu = 20, size = 1', 'mu = 20, size = 10'), eacl
  ggplot(aes(x=x, fill = params)) + geom_bar() + theme_bw() +
  facet_wrap(.~params) + theme(legend.position = 'none')
```



Count Regression

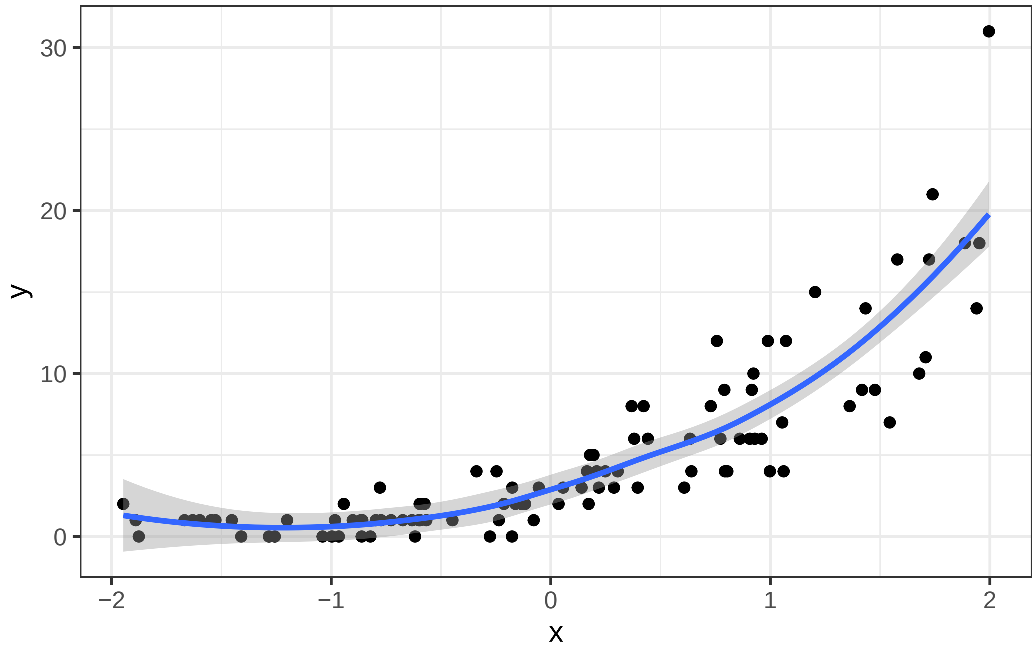
Recall that a GLM has three parts: random component, systematic component, and link function.

So with Poisson regression, it looks like this

$$
\begin{aligned}
y &\sim Poisson(\mu) \\
\mu &= \exp(\beta_0 + \beta_1 x + ...) \\
\log(\mu) &= \beta_0 + \beta_1 x + ...
\end{aligned}
$$

Let's generate data with one continuous predictor

```r
n <- 100
x <- runif(n, -2, 2)

beta0 <- 1

beta1 <- 1

mu <- exp( beta0 + beta1 * x)

y <- rpois(n, mu)

pois_reg <- tibble(y = y, x = x, mu = mu)


pois_reg |>
  ggplot(aes(y = y, x = x)) +
  geom_point() +
  geom_smooth(method = 'loess', formula = 'y ~ x') +
  theme_bw()
```



Fit a generalized linear model to your data.

```
pois_fit <- glm(y ~ x, data = pois_reg, family = poisson)
summary(pois_fit)
```

```

Call:
glm(formula = y ~ x, family = poisson, data = pois_reg)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.02489    0.07004   14.63   <2e-16 ***
x            1.00876    0.05440   18.54   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 516.58  on 99  degrees of freedom
Residual deviance:  95.27  on 98  degrees of freedom
AIC: 378.17
```

```
Number of Fisher Scoring iterations: 5
```

```
fit_line <- tibble(x = x) |>
  mutate(y = exp(coef(pois_fit)[1] + coef(pois_fit)[2] * x))
```

Add the regression fit line to your figure

```
pois_reg |>
  ggplot(aes(y = y, x = x)) +
  geom_point() +
  geom_smooth(method = 'loess', formula = 'y ~ x') +
  theme_bw() +
  geom_line(data = fit_line, color = 'red')
```