Lesson 10

Clark

https://www.youtube.com/watch?v=l2w3rkRM36E

https://www.youtube.com/watch?v=4kpDg7MjHps

How is the probability of failure in a given O-ring related to the launch temperature?

In the 23 previous shuttle missions for which data exists, some evidence of damage due to blow by and erosion was recorded on some O-rings. For each mission, we know the number of O-rings out of six showing some damage and the launch temperature.

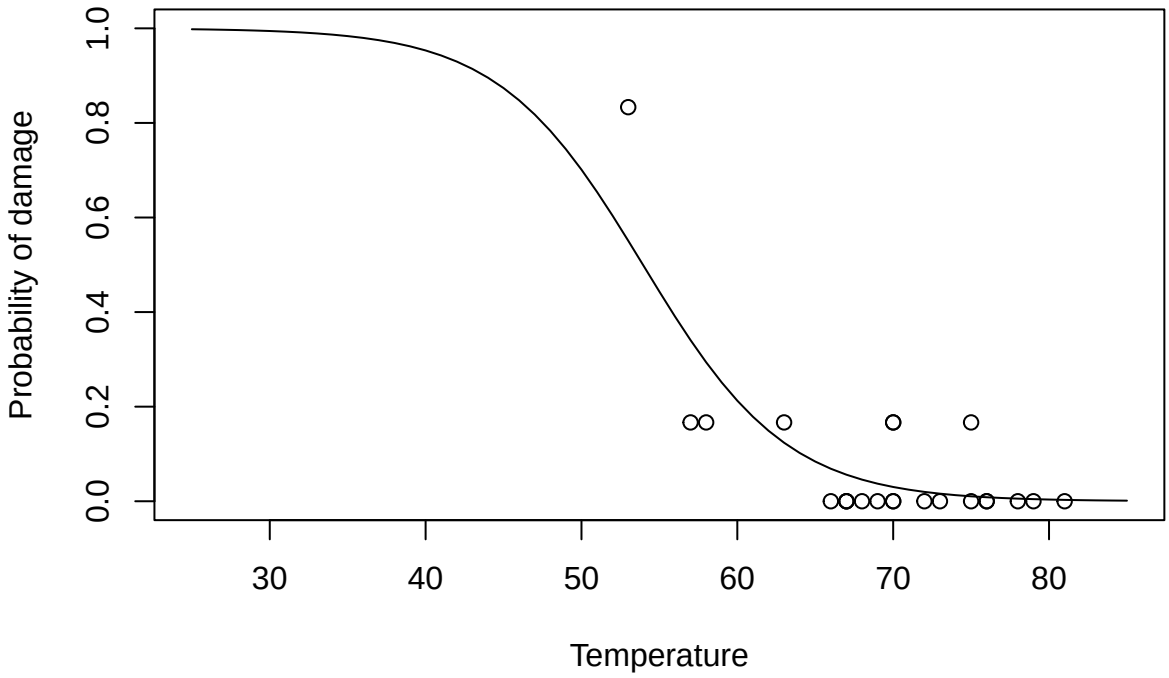
```
library(faraway)
library(tidyverse)

data(orings)

orings %>% glimpse()
```

```
## Rows: 23
## Columns: 2
## $ temp   <dbl> 53, 57, 58, 63, 66, 67, 67, 67, 68, 69, 70, 70, 70, 70, 72, 73,~
## $ damage <dbl> 5, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, ~
```

If we let Y_i be the number of defective O-rings, what is the distribution of Y_i?

This is an example of grouped data. We still have a logistic regression model, but instead of looking at individual rows of success/failure, each observational unit is a group of six trials.

```
logitmod <- glm(cbind(damage,6-damage) ~ temp, family=binomial, orings)
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim = c(0,1),
     xlab="Temperature", ylab="Probability of damage")
x <- seq(25,85,1)
lines(x,ilogit(11.6630-0.2162*x))
```



As our model is grouped, we can examine goodness of fit by looking at the Deviance. Here we have:

```
1-pchisq(logitmod$deviance,df.residual(logitmod))
```

```
## [1] 0.7164099
```

Therefore, we have no evidence of lack of fit. Note though, that this doesn't tell us our model is better than any other model, only that there is no evidence that our model does not fit our data.

Note that there is nothing to stop us form doing:

```
erings <- with(orings, data.frame(temp=rep(temp,each=6), damage=as.vector(
  sapply(orings$damage, function(x) rep(c(0,1), times=c(6-x,x)))
)))
head(erings)
```

```
##   temp damage
## 1   53      0
## 2   53      1
## 3   53      1
## 4   53      1
## 5   53      1
## 6   53      1
```

```r
emod <- glm(damage ~ temp, family = binomial("logit"), erings)
summary(emod)
```

```
##
## Call:
## glm(formula = damage ~ temp, family = binomial("logit"), data = erings)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 11.66299    3.29616   3.538 0.000403 ***
## temp        -0.21623    0.05318  -4.066 4.77e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 76.745  on 137  degrees of freedom
## Residual deviance: 54.759  on 136  degrees of freedom
## AIC: 58.759
##
## Number of Fisher Scoring iterations: 6
```

```r
summary(logitmod)
```

```
##
## Call:
## glm(formula = cbind(damage, 6 - damage) ~ temp, family = binomial,
##     data = orings)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 11.66299    3.29626   3.538 0.000403 ***
## temp        -0.21623    0.05318  -4.066 4.78e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 38.898  on 22  degrees of freedom
## Residual deviance: 16.912  on 21  degrees of freedom
## AIC: 33.675
##
## Number of Fisher Scoring iterations: 6
```

We see that the results are the same, HOWEVER, with the ungrouped data we no longer can conduct a goodness of fit test (we would have to do a Hosmer-Lemeshow type test)

An alternative to the Deviance is to compute Pearson's χ^2 statistic. Which is:

This should look familiar from the analysis of categorical data you did in MA376. We note here that the degrees of freedom, typically, for a statistic of this form are the number of rows (minus 1) times the number of columns.

The easy way to compute this is:

```
sum(residuals(logitmod, type = "pearson")^2)
```

```
## [1] 28.06738
```

or

```
phat = predict(logitmod,type="response")
n = 6
y = orings$damage
numerator = (y-n*phat)^2
denominator = n*phat*(1-phat)
sum(numerator/denominator)
```

```
## [1] 28.06738
```

This is just another goodness of fit test.

Now, let's say we fail the test, meaning we reject our deviance test (or our Pearson's χ^2 test). What could have happened?

As we build out models we want to test the simplest thign first. The simplest thing is we don't have the right X_i terms in our model. We also might have outliers, or extremely sparse data. Maybe though, we've checked (or considered) all of these and our data still doesn't fit our model well. In this case, it *could* be that our data are overdispersed.

Recall, for a Binomial distribution our $V(\mu)$ function was:

This means that our variance must link to mean in a set manner. Perhaps this isn't the case.

Let's consider the following dataset, we have: oxes of trout eggs were buried at five different stream locations and retrieved at 4 different times. The number of surviving eggs was recorded. The box was not returned to the stream.

Perhaps we build out the following model:

```
bmod <- glm(cbind(survive,total-survive) ~ location + period,
            family=binomial,troutegg)

summary(bmod)
```

```
##
## Call:
## glm(formula = cbind(survive, total - survive) ~ location + period,
##     family = binomial, data = troutegg)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   4.6358     0.2813  16.479  < 2e-16 ***
## location2    -0.4168     0.2461  -1.694   0.0903 .
## location3    -1.2421     0.2194  -5.660 1.51e-08 ***
## location4    -0.9509     0.2288  -4.157 3.23e-05 ***
## location5    -4.6138     0.2502 -18.439  < 2e-16 ***
## period7      -2.1702     0.2384  -9.103  < 2e-16 ***
## period8      -2.3256     0.2429  -9.573  < 2e-16 ***
## period11     -2.4500     0.2341 -10.466  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1021.469  on 19  degrees of freedom
## Residual deviance:   64.495  on 12  degrees of freedom
## AIC: 157.03
##
## Number of Fisher Scoring iterations: 5
```

We can look at:

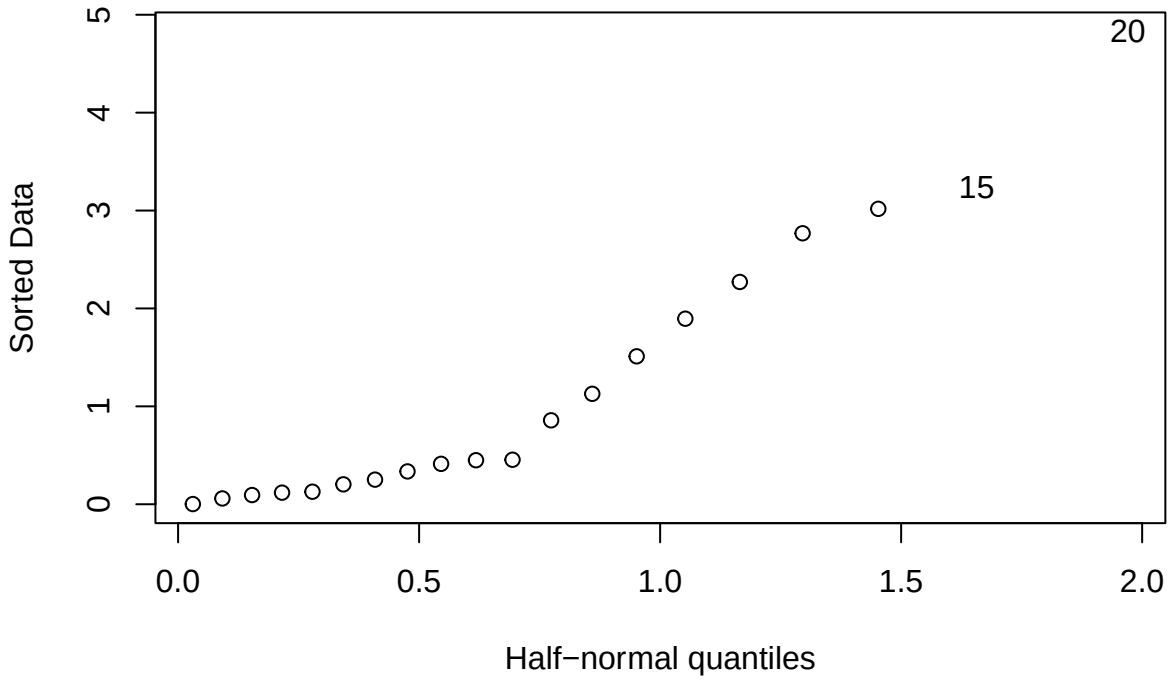
```
1-pchisq(64.5,12)
```

```
## [1] 3.372415e-09
```

Which shows evidence of lack of fit.

First thing to do is to check to ensure there's no good reason for it. Are we missing some covariates? Here we don't have anything else to consider. Are there any outliers?

```
halfnorm(residuals(bmod))
```



Nothing obvious.

Once we've done this we may consider a quasi-binomial GLM. That is, a binomial GLM with an additional dispersion parameter. First we note that our estimate of ϕ is:

```
sum(residuals(bmod,type="pearson")^2)/12
```

```
## [1] 5.330322
```

The quasi-bimomial is a way for us to create a binary regression model without forcing $\phi = 1$.

To do this, we need to abandon our previous way of fitting our GLMs. Recall that previously we had used the log-likelihood. Now, instead we are going to use the log *quasi-likelihood*. That is, we are going to find functions that behave similar to the log-likelihodo and maximize them. We first define

With this, we have something that's almost a likelihood. We next define Q_i as:

Then the full log quasi-likelihood is:

We will go through this more in later lessons. FOr now it suffices to know that the quasi-binomial is NOT fit via maximum likelihood estimates but it does allow us an additional free parameter that enables us to fit situations where the binomial struggles

```
q_bmod <- glm(cbind(survive,total-survive) ~ location + period,
              family=quasibinomial,troutegg)

summary(q_bmod)
```

```
##
## Call:
## glm(formula = cbind(survive, total - survive) ~ location + period,
##     family = quasibinomial, data = troutegg)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.6358     0.6495   7.138 1.18e-05 ***
## location2    -0.4168     0.5682  -0.734 0.477315
## location3    -1.2421     0.5066  -2.452 0.030501 *
## location4    -0.9509     0.5281  -1.800 0.096970 .
## location5    -4.6138     0.5777  -7.987 3.82e-06 ***
## period7      -2.1702     0.5504  -3.943 0.001953 **
## period8      -2.3256     0.5609  -4.146 0.001356 **
## period11     -2.4500     0.5405  -4.533 0.000686 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 5.330358)
##
##     Null deviance: 1021.469  on 19  degrees of freedom
## Residual deviance:   64.495  on 12  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Note here we no longer have an AIC. Why?

What do we see here? Any idea why?

```
confint(bmod)
```

```
## Waiting for profiling to be done...
```

```
##                  2.5 %      97.5 %
## (Intercept)  4.1010096  5.20467296
```

```
## location2   -0.9057713  0.06229966
## location3   -1.6852396 -0.82256141
## location4   -1.4101732 -0.51073675
## location5   -5.1215188 -4.13917831
## period7     -2.6490979 -1.71337087
## period8     -2.8134876 -1.85999146
## period11    -2.9215121 -2.00276803
```

```
confint(q_bmod)
```

```
## Waiting for profiling to be done...

##                 2.5 %     97.5 %
## (Intercept)  3.448222  6.0058518
## location2   -1.578567  0.6907938
## location3   -2.315039 -0.2980182
## location4   -2.055369  0.0488014
## location5   -5.844323 -3.5608927
## period7     -3.315557 -1.1429485
## period8     -3.491540 -1.2784566
## period11    -3.581212 -1.4489377
```

```
prds <- predict(bmod,se.fit=TRUE)

q_prds <- predict(q_bmod,se.fit=TRUE)


prds$se.fit
```

```
##         1         2         3         4         5         6         7         8
## 0.2813161 0.2641014 0.2367007 0.2474624 0.1834887 0.2162603 0.1926780 0.1536563
##         9        10        11        12        13        14        15        16
## 0.1721610 0.1952807 0.2167479 0.1927754 0.1545731 0.1699225 0.2029157 0.1982736
##        17        18        19        20
## 0.1812000 0.1466056 0.1557885 0.1919134
```

```
q_prds$se.fit
```

```
##         1         2         3         4         5         6         7         8
## 0.6494904 0.6097458 0.5464843 0.5713304 0.4236308 0.4992925 0.4448465 0.3547550
##         9        10        11        12        13        14        15        16
## 0.3974780 0.4508557 0.5004182 0.4450715 0.3568718 0.3923097 0.4684830 0.4577656
##        17        18        19        20
## 0.4183468 0.3384767 0.3596776 0.4430813
```

So, we have a trade-off here, our model is more appropriate for our data but we have less precision in our parameters. We will talk more in depth about quasi-likelihood in later classes.