Lsn 24

Clark

Admin

Last class we talked about modifying the right hand side of our regression model. That is, we could fit the class of models:

Where $f(x_i)$ that we considered was the class of polynomial functions. Another class of regression models is fitting:

Why would we want to do this? Well, we might do it because we have a reason to believe that this is the underlying relationship. Another common reason is that our assumptions are violated. For instance, speed and stopping distance look like:

```
stopping.dat<-read_table2("http://www.isi-stats.com/isi2/data/stopping.txt")%>%drop_na()
```

```
## Parsed with column specification:
## cols(
##   speed = col_integer(),
##   stoppingdistance = col_integer()
## )
```

```
stopping.dat %>% ggplot(aes(x=speed,y=stoppingdistance))+
  geom_point()
```

Here we see that a model relating speed to stopping distane linearly might not be appropriate. Why?

To account for this we might consider transforming y. One common transformation is $\log(y)$ (Note: From here on out in life, when someone writes log assume that it is natural log)

Why $\log(y)$? This comes from the delta method in statistics. The assumption we are making when we use the log transformation is that the variance of y is increasing as the expected value of y increases. That is:

The delta method says that any transformation of a random variable can be expressed as:

Similarly $\sqrt{y}$ can be used if we have y that has variance of:

So, any power transformation can be done if we want to correct or stabilize our variance and our variance is a function of our mean.

So why not transform?

Well, it changes the relationship between x_i and y_i. Note that previously we had assumed x_i and y_i had a linear relationship. Now, let's take $\log(y_i)$ and fit a regression model. The model is now:

So now the relationship between x_i and y_i becomes:

Overall, my recommendation is, if I don't care about exploring a linear relationship between x_i and y_i then transform away in order to fix assumptions. If we DO care about the linear relationshp, then we shouldn't do this.

Here we could do:

```
stopping.dat=stopping.dat %>% mutate(log.stop=log(stoppingdistance))
log.lm<-lm(log.stop~speed,data=stopping.dat)

log.lm %>% ggplot(aes(x=.fitted,y=.resid))+geom_point()
```

If we are satisfied with this we could find:

```
predict(log.lm,data.frame(speed=4),interval="prediction")
```

```
##        fit      lwr      upr
## 1 1.854145 1.104573 2.603718
```

But remember that this isn't a PI for stopping. It is for log(stopping), so our 95 % PI for stopping is

```
exp(1.1)
```

```
## [1] 3.004166
```

```
exp(2.6)
```

```
## [1] 13.46374
```

Let's see what happens though if we choose a different transformation:

```
stopping.dat=stopping.dat %>% mutate(sq.stop=sqrt(stoppingdistance))
sq.lm<-lm(sq.stop~speed,data=stopping.dat)

sq.lm %>% ggplot(aes(x=.fitted,y=.resid))+geom_point()
```

Fit looks better

```
predict(sq.lm,data.frame(speed=4),interval="prediction")
```

```
##        fit       lwr      upr
## 1 1.928557 0.4527392 3.404374
```

Again, this is for square root of stopping time, so to find actual interval we need:

```
.452^2
```

```
## [1] 0.204304
```

```
3.4^2
```

```
## [1] 11.56
```

So certainly the transformation matters. How do we know that we have the *right* transformation? Well, we cannot do:

```
anova(sq.lm,log.lm)
```

```
## Warning in anova.lmlist(object, ...): models with response '"log.stop"'
## removed because response differs from model 1
```

```
## Analysis of Variance Table
##
## Response: sq.stop
##           Df Sum Sq Mean Sq F value    Pr(>F)
## speed      1 386.06  386.06  746.22 < 2.2e-16 ***
## Residuals 61  31.56    0.52
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In fact it gives us a warning. Our models are NOT nested, so statistics doesn't help us here.

One way is to note that all of the transformations are special cases of what are known as box-cox transformations.

So we can find the value of λ that maximizes the log-likelihood of this.

```
library(MASS)
```

```
##
```

```
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
boxcox(stoppingdistance~speed,data=stopping.dat)
```

Here $\lambda = 0$ is log transformation and $\lambda = .5$ is square root, $\lambda = 1$ is no transformation. So it looks like $\lambda \approx .5$ would be preferable here.

So the model would be:

Note here there is not a nice clean linear or multiplicative relationship between x_i and y_i, but there is **a** relationship. If we want a predictive model this would suffice but if we want to explore a linear relationship between x_i and y_i this would not be appropriate.