Lecture 14: Other Models

Taylor Arnold

lm()

So far, we have seen how to use the function `lm_basic` to build linear regression models.

As I probably mentioned before, this is just a soft wrapper around the base R function `lm`, with the following tweaks:

- you are forced to supply a data frame
- the summary table produces confidence intervals rather than standard errors

I don't use this function in my own work, but find it easier for teaching.

lm()

One major benefit of the "real" `lm` is that there are many more options available for calling it, such as including sample weights.

glm()

There are many other functions in R that have similar calling mechanisms to `lm` but run different underlying models.

For example, `glm` fits generalized linear models. These can be used, amongst other things, for fitting a model to a binary response:

```
df <- data_frame(y = c(0,0,0,0,1,1,1,1), x = rnorm(8))
model <- glm(y ~ x, data = df, family = binomial())
```

glm()

```
summary(model)
```

```
##
## Call:
## glm(formula = y ~ x, family = binomial(), data = df)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.28128  -1.13899   0.01232   1.15536   1.24458
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.02157    0.71807  -0.030    0.976
## x            0.16454    0.88998   0.185    0.853
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

rlm()

In the **MASS** package (included with all standard R installations) is the `rlm` function for fitting robust linear regression:

```
library(MASS)
x <- rnorm(100)
y <- 1 + x + rnorm(100, sd = 0.2)
y[50] <- 1e4
model_lm <- lm(y ~ x)
model_rlm <- rlm(y ~ x)
```

rlm()

```
summary(model_lm)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -433.7 -184.2  -99.7  -15.0 9734.3
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    117.1      100.4   1.166    0.246
## x             -142.3      110.9  -1.283    0.203
##
## Residual standard error: 996.6 on 98 degrees of freedom
## Multiple R-squared:  0.01651,   Adjusted R-squared:  0.006479
```

rlm()

```
summary(model_rlm)
```

```
##
## Call: rlm(formula = y ~ x)
## Residuals:
##       Min          1Q      Median          3Q         Max
## -4.573e-01 -1.245e-01 -2.746e-02  1.196e-01  1.000e+04
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept)  0.9980  0.0227    44.0345
## x            0.9965  0.0250    39.8049
##
## Residual standard error: 0.1934 on 98 degrees of freedom
```

other models

If you have a need for a specific model, you can usually find an R package that support it. In most cases, the model will roughly resemble calling `lm`.

Some common examples you may run into:

- `gam::gam` for generalized additive models
- `nls` for non-linear regression
- `lme4::lmer` for mixed effects models
- `quantreg::qr` for quantila regression
- `glmnet::glmnet` for the generalized elastic net
- `randomforest::randomforest` for random forest classifications
- `forcast::auto.arima` for modeling time series