BIFX 553 - Discussion 5

Randy Johnson

February 16, 2017

Tests of Association

Single parameter

We've already seen and discussed p-values in regression output, but we will discuss them in more detail here. Our favorite model is currently

$$nodes_i = \beta_0 + age_i * \beta_1 + size_i * \beta_2 + grade_i * \beta_3 + \varepsilon_i.$$

```
# load the GBSG data from Discussion 2
load('../1-26/gbsg.RData')

# revisit our model
full.model <- lm(lnodes ~ age + size + grade + meno + lpgr + ler + hormon, data = gbsg)
```

```
## Error in eval(expr, envir, enclos): object 'lnodes' not found
```

```
tidy(full.model)
```

```
## Error in tidy(full.model): object 'full.model' not found
```

Given this model, what are the statistically significantly associated predictors of the number of nodes? How would you describe these associations? Can we remove any variables from the model without loosing information?

Multiple degree of freedom tests

That last question is perhaps best answered with a multiple degree of freedom test. Lets say that we want to check our model against a model without `age`, `grade`, `pgr`, `er` and the `size:grade` interaction. We can do this with the `anova` function in R.

```
# this is our alternate model
alt.model <- update(full.model, . ~ . - age - lpgr - ler - hormon)
```

```
## Error in update(full.model, . ~ . - age - lpgr - ler - hormon): object 'full.model' not found
```

```
# check if we are loosing a significant amount of information if we stick with the alternate model
anova(full.model, alt.model)
```

```
## Error in anova(full.model, alt.model): object 'full.model' not found
```

It appears as if we could trimg the `full.model` down a bit in favor of the `alt.model`. Does this fit well with our disease model?.

Missing Data

Types of missing data

- Missing completely at random: no factors relating to the samples (measured or not) influenced which data are missing. Example: A study ends prematurely, and some data are not able to be collected,

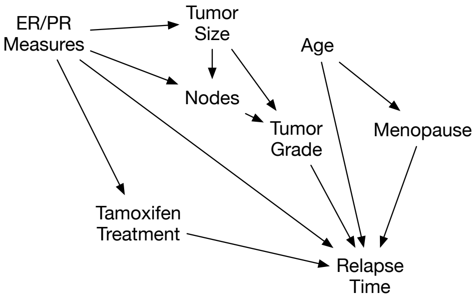

Figure 1: DAG summarizing our disease model.

independent of any sample characteristics.

- Missing at random: some important factors may have influenced which data are missing, but the probablility of missingness is a function of variables that were measured. Example: Individuals with depression may be more likely to be lost to followup, resulting in missing data. As long as loss to followup isn't related to the variable that is missing (e.g. number of cigarettes smoked each week during the study), we can assume that the number of cigarettes smoked by individuals we did observe is representative of the number of cigarettes smoked by individuals we didn't observe.
- Informative missing: missing data are biased in some way by other confounding variables. This results in estimates that are higher or lower than they should be. This is often nearly impossible to detect without some outside information (e.g. experience with past studies or knowledge of the population under study). Example: some unmeasured counfounding variable(s) influences heavy smokers in the treatment group to drop out of the study at a higher rate than individuals who smoke less.

Problems stemming from missing data

Always characterize missingness of data. Ask questions like:

- What is the rate of missingness in each group?
- Are there any factors in our disease model that would cause an individual to have missing data?
- What other relationships between observed data and missingness exist?

Informative missingness can cause unexpected problems, including false associations. Example: In the previous presidential election, the "undecided" voters (i.e. likely voters with missing data) voted disproportionately for President Trump. This failed assumption resulted in unreliable polling leading up to the election.

Dealing with missing data

- Ignore missing data

```
set.seed(239847)
n <- 100
# generate a dataset with a lot of missing data
dat <- data_frame(x1 = rnorm(n),
                  x2 = rnorm(n),
                  g = rbinom(n, 1, .5),
                  y = x1 + x2 + x1*x2 + (g == 1) + rnorm(n)) %>%
       prodNA()
dat
```

```
## # A tibble: 100 × 4
##              x1         x2     g           y
##           <dbl>      <dbl> <int>       <dbl>
## 1   0.43788108  2.0317248    NA  3.4269158
## 2   0.51206258 -0.4317878     1 -0.3195028
## 3   0.37422403  0.2143055     0  1.0164943
## 4   0.59381328  1.0086273     1  1.6125397
## 5   1.97367993  0.2218173     1  1.4531170
## 6  -0.73490075  0.3132395     0 -0.5675547
## 7   0.86301637 -0.2132996     0  0.7172030
## 8  -1.28152664 -0.2756235     0 -2.5823621
## 9   1.14205170  1.0586741     1  4.9751146
## 10 -0.09092382  1.8772961     1  1.0887537
## # ... with 90 more rows
```

```
# look at the relationship between x and y by g
lm(y ~ x1*x2 + g, data = dat) %>%
  tidy()
```

```
##          term   estimate std.error statistic      p.value
## 1 (Intercept) 0.04631757 0.1777562  0.260568 7.952892e-01
## 2          x1 0.90845393 0.1310362  6.932848 2.809523e-09
## 3          x2 0.90704660 0.1392570  6.513474 1.483615e-08
## 4           g 0.70943829 0.2655294  2.671788 9.626272e-03
## 5       x1:x2 1.48408561 0.1522519  9.747566 3.969240e-14
```

- Replace missing data with the group mean

```
# replace
dat <- mutate(dat,
              mx1 = ifelse(is.na(x1), mean(dat$x1, na.rm = TRUE), x1),
              mx2 = ifelse(is.na(x2), mean(dat$x2, na.rm = TRUE), x2),
              mg = ifelse(is.na(g), mean(dat$g, na.rm = TRUE), g))

# look at the relationship between x and y by g
lm(y ~ mx1*mx2 + g, data = dat) %>%
  tidy()
```

```
##          term  estimate std.error statistic      p.value
## 1 (Intercept) 0.1157842 0.1746073 0.6631121 5.091650e-01
## 2         mx1 1.0477377 0.1377082 7.6083911 4.710378e-11
## 3         mx2 0.9235718 0.1485312 6.2180344 2.152961e-08
## 4           g 0.6426178 0.2562871 2.5074140 1.418613e-02
## 5     mx1:mx2 1.4630055 0.1647692 8.8791187 1.517980e-13
```

- Impute

```
imp <- select(dat, -mx1, -mx2, -mg) %>%
       as.data.frame() %>% # won't accept a tibble
       missForest()
```

```
##   missForest iteration 1 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
##   missForest iteration 2 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
##   missForest iteration 3 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
##   missForest iteration 4 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
##   missForest iteration 5 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
##   missForest iteration 6 in progress...
```

```
## Warning in randomForest.default(x = obsX, y = obsY, ntree = ntree, mtry =
## mtry, : The response has five or fewer unique values. Are you sure you want
## to do regression?
```

```
## done!
```

```r
lm(y ~ x1*x2 + g, data = imp$ximp) %>%
  tidy()
```

```
##          term   estimate std.error  statistic      p.value
## 1 (Intercept) 0.01593081 0.1382580  0.1152252 9.085098e-01
## 2          x1 0.98954739 0.1081554  9.1493097 1.087661e-14
## 3          x2 0.99484627 0.1011951  9.8309763 3.794908e-16
## 4           g 0.73097516 0.2064832  3.5401185 6.213726e-04
## 5       x1:x2 1.40140924 0.1144621 12.2434315 3.023447e-21
```