nobel_winners_infer

王敏杰

2019-06-01

一个探索性分析，四个假设检验的方法。方法来源于 [https://www.andrewheiss.com/blog/2019/01/29/diff-means-half-dozen-ways/]

1 导入数据

这是 2019 年 tidytuesday 的一个关于诺贝尔奖获得者的数据集

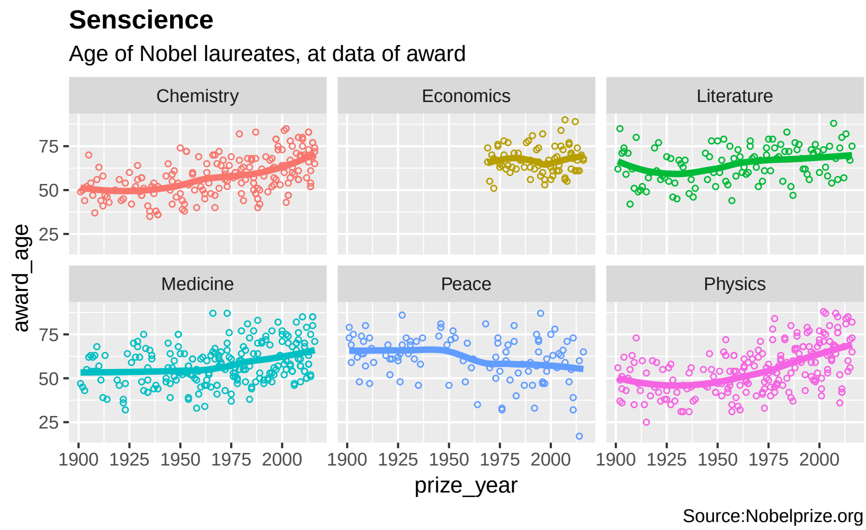
```
nobel_winners <- read_csv(here::here("2019-05-14", "nobel_winners.csv"))
```

```
# df <- here::here("2019-05-14") %>%
#   dir_ls(regexp = "\\.csv$") %>%
#   map_dfr(read_csv, .id = "source")
```

2 探索性分析

我们获取获奖者在获奖时的年龄

```
# 分类汇总
nobel_winners %>%
  mutate(award_age = prize_year - year(birth_date)) %>%
  ggplot(aes(x = prize_year, y = award_age, color = category)) +
  geom_point(shape = 1, size = 1) +
  geom_smooth(se = FALSE, lwd = 1.5) +
  facet_wrap(vars(category)) +
  theme(
    plot.title = element_text(face = "bold"),
    legend.position = "none"
  ) +
  labs(
    title = "Senscience",
    subtitle = "Age of Nobel laureates, at data of award",
    caption = "Source:Nobelprize.org"
  )
```



这里我们只关注物理学和和平奖这两大类的诺贝尔奖

```
df <- nobel_winners %>%
  mutate(award_age = prize_year - year(birth_date)) %>%
  select(category, award_age) %>%
  filter(category %in% c("Physics", "Peace")) %>%
  filter(!is.na(award_age))

df
#> # A tibble: 323 x 2
#>    category award_age
#>    <chr>        <dbl>
#>  1 Peace           73
#>  2 Peace           79
#>  3 Physics         56
#>  4 Peace           69
#>  5 Peace           59
#>  6 Physics         49
#>  7 Physics         37
#>  8 Peace           75
#>  9 Physics         51
#> 10 Physics         44
#> # ... with 313 more rows
```

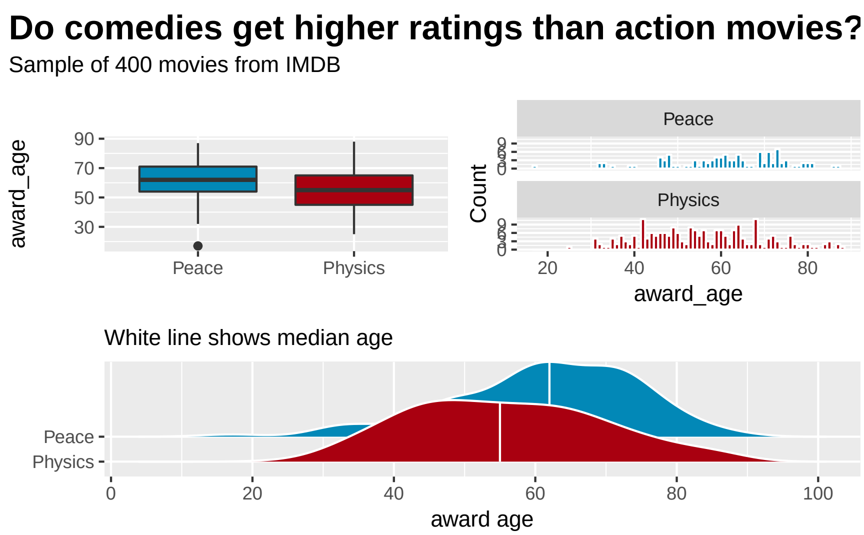
```
eda_boxplot <- df %>%
  ggplot(aes(x = category, y = award_age, fill = category )) +
  geom_boxplot() +
  scale_fill_manual(values = c("#0288b7", "#a90010"), guide = FALSE) +
```

```r
  #scale_y_continuous(breaks = seq(1, 10, 1)) +
  labs(x = NULL, y = "award_age")
#eda_boxplot
```

```r
eda_histogram <- df %>%
  ggplot(mapping = aes(x = award_age, fill = category )) +
  geom_histogram(binwidth = 1, color = "white") +
  scale_fill_manual(values = c("#0288b7", "#a90010"), guide = FALSE) +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  labs(x = "award_age", y = "Count") +
  facet_wrap(vars(category), nrow = 2) +
  theme(panel.grid.major.x = element_blank())
#eda_histogram
```

```r
eda_ridges <- df %>%
  ggplot(aes(x = award_age, y = fct_rev(category), fill = category)) +
  stat_density_ridges(quantile_lines = TRUE, quantiles = 2, scale = 3, color = "white") +
  scale_fill_manual(values = c("#0288b7", "#a90010"), guide = FALSE) +
  scale_x_continuous(breaks = seq(0, 100, 20)) +
  labs(
    x = "award age", y = NULL,
    subtitle = "White line shows median age"
  )
#eda_ridges
```

```r
showtext_auto()
(eda_boxplot | eda_histogram) /
    eda_ridges +
  plot_annotation(title = "Do comedies get higher ratings than action movies?",
                  subtitle = "Sample of 400 movies from IMDB",
                  theme = theme(plot.title = element_text(face = "bold",
                                                          size = rel(1.5))))
#> Picking joint bandwidth of 4.36
```



3 获奖年龄是否有差异

我们的问题是，We are looking to see if a difference exists in the mean award_age of the two levels of the explanatory variable.

```
group_diffs <- df %>%
  group_by(category) %>%
  summarize(mean = mean(award_age),
            std_dev = sd(award_age),
            n = n()) %>%
  {.$mean[2] - .$mean[1] }
```

Yep. There's a -5.5383495 point difference in ratings. Action movies score 0.7 points lower than comedies, on average.

But how certain are we that that difference is real and not just due to sampling error? It's time for inference!

3.1 Classical frequentist t-tests

3.1.1 t-test, assuming equal variances

We can use a standard frequentist t-test to check if the group means are different. We can assume that the variances in the two groups are the same and run `t.test()`:

```
t_test_eq <-
    t.test(award_age ~ category, data = df, var.equal = TRUE) # 假定方程是相等的
t_test_eq
#>
```

```
#>  Two Sample t-test
#>
#> data:  award_age by category
#> t = 3.4286, df = 321, p-value = 0.0006859
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#>  2.360322 8.716377
#> sample estimates:
#>   mean in group Peace mean in group Physics
#>              61.38835              55.85000
```

```
t_test_eq_tidy <- tidy(t_test_eq) %>%
    mutate(estimate = estimate1 - estimate2) %>%
    select(starts_with("estimate"), everything())
t_test_eq_tidy
#> # A tibble: 1 x 10
#>   estimate1 estimate2 estimate statistic p.value parameter conf.low
#>       <dbl>     <dbl>    <dbl>     <dbl>   <dbl>     <dbl>    <dbl>
#> 1      61.4      55.8     5.54      3.43 6.86e-4       321     2.36
#> # ... with 3 more variables: conf.high <dbl>, method <chr>,
#> #   alternative <chr>
```

3.2 t-test, assuming unequal variance

We can run a t-test assuming that the two groups have unequal variances by setting `var.equal = FALSE`, or just leaving it off. I generally just do this instead of going through all the tests for equal variance.

```
t_test_uneq <-
    t.test(award_age ~ category, data = df)  # 假定方差是不等的

t_test_uneq_tidy <- tidy(t_test_uneq) %>%
    mutate(estimate = estimate1 - estimate2) %>%
    select(starts_with("estimate"), everything())
t_test_uneq_tidy
#> # A tibble: 1 x 10
#>   estimate estimate1 estimate2 statistic p.value parameter conf.low
#>      <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>
#> 1     5.54      61.4      55.8      3.49 5.79e-4      209.     2.41
#> # ... with 3 more variables: conf.high <dbl>, method <chr>,
#> #   alternative <chr>
```

3.3 Common tests are linear models

Physics as baseline

[https://lindeloev.github.io/tests-as-linear/]

```
df %>%
    mutate(category = fct_rev(category)) %>%
    lm(award_age ~ 1 + category, data = .) %>%
    broom::tidy()
#> # A tibble: 2 x 5
#>   term           estimate std.error statistic   p.value
#>   <chr>             <dbl>     <dbl>     <dbl>     <dbl>
#> 1 (Intercept)       55.8      0.912     61.2  4.23e-179
#> 2 categoryPeace      5.54     1.62       3.43 6.86e-  4
```

3.4 infer: Simulation-based tests

[https://allendowney.blogspot.com/2016/06/there-is-still-only-one-test.html]

```
df %>%
  specify(formula = award_age ~ category) %>%
  calculate(stat = "t",                 # 这个 t 是什么意思
            order = c("Peace", "Physics")
  )
#> # A tibble: 1 x 1
#>    stat
#>   <dbl>
#> 1  3.49
```

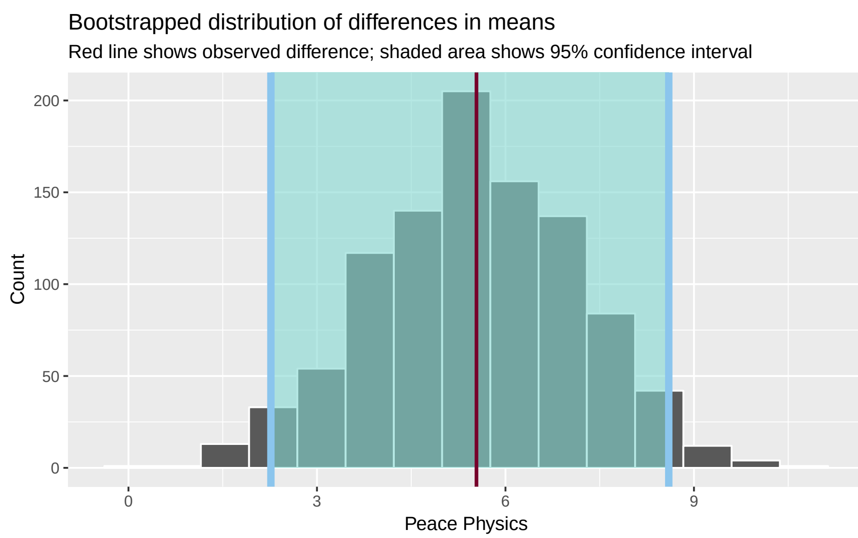
这是的 t 是 t.test，用在常规的情形。但这里，模拟的情况下, 不适用了。?

我们要用 `stat = "diff in means"` First we calculate the difference in means in the actual data:

```
diff_means <- df %>%
  specify(formula = award_age ~ category) %>%
  calculate(stat = "diff in means",      # 这个 t 与 diff in means 区别?
            order = c("Peace", "Physics")
  )
diff_means
#> # A tibble: 1 x 1
#>    stat
#>   <dbl>
#> 1  5.54
```

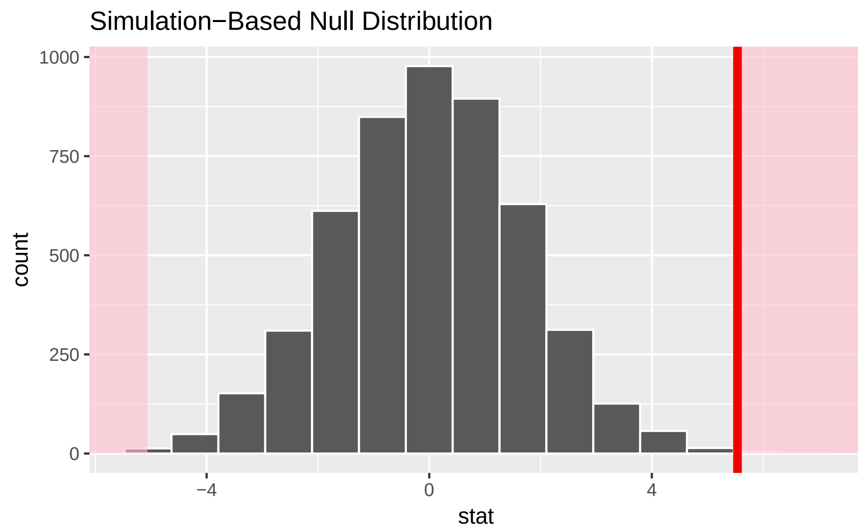
Then we can generate a bootstrapped distribution of the difference in means based on our sample and calculate the confidence interval:

```
boot_means <- df %>%
  specify(award_age ~ category) %>%
  generate(reps = 1000, type = "bootstrap") %>%
  calculate("diff in means", order = c("Peace", "Physics"))

boostrapped_confint <- boot_means %>% get_confidence_interval()

boot_means %>%
  visualize() +
  shade_confidence_interval(boostrapped_confint,
                            color = "#8bc5ed", fill = "#85d9d2") +
  geom_vline(xintercept = diff_means$stat, size = 1, color = "#77002c") +
  labs(title = "Bootstrapped distribution of differences in means",
       x = "Peace Physics", y = "Count",
       subtitle = "Red line shows observed difference; shaded area shows 95% confidence interval")
```




```
category_diffs_null <- df %>%
  specify(formula = award_age ~ category) %>%
  hypothesize(null = "independence") %>%
  generate(reps = 5000, type = "permute")  %>%
  calculate(stat = "diff in means",
            order = c("Peace", "Physics")
  )
```

```r
category_diffs_null %>%
  visualize() +
  shade_p_value(obs_stat = diff_means, direction = "both")
```




```r
category_diffs_null %>%
  get_pvalue(obs_stat = diff_means, direction = "both")
#> # A tibble: 1 x 1
#>   p_value
#>     <dbl>
#> 1 0.00120
```

Because the p-value is so small, it passes pretty much all evidentiary thresholds ($p < 0.05$, $p < 0.01$, etc), so we can safely say that there's a difference between the two groups. Action movies are rated lower, on average, than comedies

3.5 Bayesian regression

3.5.1 Regression, assuming equal variances

brms 的方法

```r
brms_eq <- brm(
  bf(award_age ~ category),
  data = mutate(df, category = fct_rev(category)),
  prior = c(
    set_prior("normal(57, 5)", class = "Intercept"),
    set_prior("normal(5.5, 1)", class = "b")
  ),
  chains = 4, iter = 4000, warmup = 2000, seed = 1024
```

```
)
```

```
brms_eq_tidy <-
  tidyMCMC(brms_eq,
    conf.int = TRUE, conf.level = 0.95,
    estimate.method = "median", conf.method = "HPDinterval"
  )
brms_eq_tidy
#> # A tibble: 3 x 5
#>   term             estimate std.error conf.low conf.high
#>   <chr>               <dbl>     <dbl>    <dbl>     <dbl>
#> 1 b_Intercept         55.8      0.806    54.3      57.4
#> 2 b_categoryPeace      5.52     0.848     3.90      7.22
#> 3 sigma               13.5      0.528    12.5      14.6
```

```
broom.mixed::tidy(brms_eq)
#> Warning in checkMatrixPackageVersion(): Package version inconsistency detected.
#> TMB was built with Matrix version 1.2.15
#> Current Matrix version is 1.2.16
#> Please re-install 'TMB' from source using install.packages('TMB', type = 'source') or ask CRAN for
#> # A tibble: 3 x 8
#>   effect  component group   term      estimate std.error conf.low conf.high
#>   <chr>   <chr>     <chr>   <chr>        <dbl>     <dbl>    <dbl>     <dbl>
#> 1 fixed   cond      <NA>    (Interce~    55.8      0.806    54.2      57.4
#> 2 fixed   cond      <NA>    category~     5.52     0.848     3.87      7.19
#> 3 ran_pa~ cond      Residu~ sd__Obse~    13.6      0.528    12.6      14.7
```

3.5.2 Regression, assuming unequal variances

```
brms_uneq <- brm(
  bf(award_age ~ category, sigma ~ category),
  data = mutate(df, category = fct_rev(category)),
  prior = c(
    set_prior("normal(57, 5)", class = "Intercept"),
    set_prior("normal(5.5, 1)", class = "b"),
    set_prior("cauchy(0, 1)", class = "b", dpar = "sigma")
  ),
  chains = 4, iter = 4000, warmup = 2000, seed = 1024
)
```

```r
brms_uneq_tidy <-
  tidyMCMC(brms_uneq, conf.int = TRUE, conf.level = 0.95,
           estimate.method = "median", conf.method = "HPDinterval")
brms_uneq_tidy
#> # A tibble: 4 x 5
#>   term                 estimate std.error conf.low conf.high
#>   <chr>                   <dbl>     <dbl>    <dbl>     <dbl>
#> 1 b_Intercept           55.8       0.787    54.3      57.4
#> 2 b_sigma_Intercept       2.62      0.0476    2.53      2.72
#> 3 b_categoryPeace         5.50      0.842     3.92      7.20
#> 4 b_sigma_categoryPeace  -0.0519    0.0849   -0.216     0.115
```

For mathy reasons (again, see Matti Vourre's post), the sigma terms are on a log scale, so we need to exponentiate them back to the scale of the data.

```r
brms_uneq_tidy %>%
  mutate_at(vars(estimate, std.error, conf.low, conf.high),
            funs(ifelse(str_detect(term, "sigma"), exp(.), .)))
#> Warning: funs() is soft deprecated as of dplyr 0.8.0
#> please use list() instead
#>
#>   # Before:
#>   funs(name = f(.))
#>
#>   # After:
#>   list(name = ~ f(.))
#> This warning is displayed once per session.
#> # A tibble: 4 x 5
#>   term                 estimate std.error conf.low conf.high
#>   <chr>                   <dbl>     <dbl>    <dbl>     <dbl>
#> 1 b_Intercept            55.8      0.787    54.3       57.4
#> 2 b_sigma_Intercept      13.8      1.05     12.6       15.1
#> 3 b_categoryPeace         5.50     0.842     3.92       7.20
#> 4 b_sigma_categoryPeace   0.949    1.09      0.806      1.12
```

4 比较各种方法

Holy cow, that's a lot of code. We can compare the output from all these different methods in a single plot. In this case, since both groups are pretty normally distributed already and there were no outliers, there isn't much variation at all in the results - all the different methods show essentially the same thing. We can legally interpret the Bayesian results using credible intervals and probabilities; with the classical t-tests, we still have to talk about null hypotheses. But in the end, the results are nearly identical (but that's definitely not always the case).

```
# Make a bunch of data frames that have three columns:
# estimate, conf.low, and conf.high

# Extract t-test results
t_test_eq_small <- t_test_eq_tidy %>%
  select(estimate, conf.low, conf.high)

t_test_uneq_small <- t_test_uneq_tidy %>%
  select(estimate, conf.low, conf.high)

# Extract simulation results
infer_simulation <- tibble(estimate = diff_means$stat,
                           conf.low = boostrapped_confint$`2.5%`,
                           conf.high = boostrapped_confint$`97.5%`)

# Extract brms regression results
brms_eq_small <- brms_eq_tidy %>%
  filter(term == "b_categoryPeace") %>%
  select(estimate, conf.low, conf.high)

brms_uneq_small <- brms_uneq_tidy %>%
  filter(term == "b_categoryPeace") %>%
  select(estimate, conf.low, conf.high)

# Put all these mini dataframes into a list column, then unnest
meta_diffs <- tribble(
  ~package, ~method, ~results,
  "t-test", "equal variances", t_test_eq_small,
  "t-test", "unequal variances", t_test_uneq_small,
  "infer", "simulation", infer_simulation,
  "brms", "equal variances", brms_eq_small,
  "brms", "unequal variances", brms_uneq_small
```

```r
) %>%
  unnest(results) %>%
  mutate(method = paste0(package, ": ", method)) %>%
  mutate(method = fct_inorder(method))

ggplot(meta_diffs, aes(x = estimate, y = fct_rev(method), color = package)) +
  geom_pointrangeh(aes(xmin = conf.low, xmax = conf.high), size = 1) +
  geom_vline(xintercept = 0, size = 1) +
  scale_color_viridis_d(option = "plasma", end = 0.9, guide = FALSE) +
  labs(x = "Mean rating for action movies mean rating for comedies",
       y = NULL, caption = "Sample of 400 movies from IMDB",
       title = "Comedies get higher ratings than action movies",
       subtitle = "Effect is roughly the same regardless of method used") +
  expand_limits(x = 0) +
  theme(plot.title = element_text(face = "bold", size = rel(1.5)))
```