Statistics in R

James McDermott

University of Galway

Statistics in R

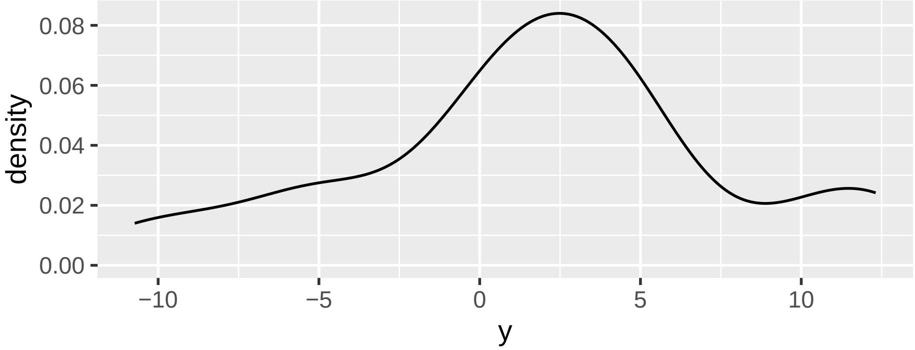
Load Tidyverse as usual

```
library(tidyverse)
```

```
## -- Attaching packages -----------------------------
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ---------------------------------- tid
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Random numbers

```
x1 <- runif(20, min=0, max=2) # random uniform with bounds
x2 <- rnorm(20) # random normal with mean 0, sd 1
y <- x1 + x2 * rnorm(20, mean=5, sd=2)
ggplot(tibble(y), aes(x=y)) + geom_density()
```



Basic statistics

```r
for (f in c(min, max, mean, median, sd, var, IQR, mad)) {
  print(f(y))
}
```

```
## [1] -10.7297
## [1] 12.31461
## [1] 1.778196
## [1] 1.797263
## [1] 5.989516
## [1] 35.87431
## [1] 5.365061
## [1] 4.464351
```

More data summaries

```
for (f in c(range, quantile, summary, fivenum)) {
  print(f(y))
}
```

```
## [1] -10.72970  12.31461
##         0%        25%        50%        75%       100%
## -10.729703  -1.026139   1.797263   4.338922  12.314614
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -10.730  -1.026   1.797   1.778   4.339  12.315
## [1] -10.729703  -1.213900   1.797263   4.675367  12.314614
```

Correlations

```r
cor(x1, y) # get the correlation
```

```
## [1] 0.03022614
```

Correlations: statistical test

```r
cor.test(x1, y) # run a test
```

```
##
##  Pearson's product-moment correlation
##
## data:  x1 and y
## t = 0.1283, df = 18, p-value = 0.8993
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4178841  0.4665071
## sample estimates:
##        cor
## 0.03022614
```

Correlations: using results

```
res = cor.test(x1, y) # save the result
names(res) # see result structure
```

```
## [1] "statistic"   "parameter"   "p.value"     "estimate"
## [6] "alternative" "method"      "data.name"   "conf.int"
```

```
R = res['statistic'] # extract values...
p = res['p.value'] # ...from the result
```

Null hypothesis significance testing

Independent 2-sample 2-sided t-test

Test whether difference in means is different from 0

```
t.test(x1, y)
```

```
##
##  Welch Two Sample t-test
##
## data:  x1 and y
## t = -0.53456, df = 19.362, p-value = 0.599
## alternative hypothesis: true difference in means is not equ
## 95 percent confidence interval:
##  -3.532283  2.093613
## sample estimates:
## mean of x mean of y
##  1.058860  1.778196
```

More t-tests

The `t.test` function also has options for:

- 1-sided tests
- paired tests
- 1-sample tests.

Regression models

The `lm` (linear model) function and variants are used for regression.

```
df = tibble(x1, x2, y)
head(df)
```

```
## # A tibble: 6 x 3
##       x1     x2      y
##    <dbl>  <dbl>  <dbl>
## 1 1.15   -0.434 -0.838
## 2 1.71   -0.132  1.01
## 3 1.18    0.835  3.98
## 4 0.0874  0.856  5.35
## 5 0.462   0.627  4.00
## 6 1.21    0.535  2.02
```

Formulas

R provides a special formula syntax involving the tilde `~`. It's used to specify a regression model. The left-hand side is the dependent variable, `y`. The right-hand side gives the independent variables, interactions, and transformations. So, `~` means something like "is modelled as".

```
y ~ x1 + x2
```

This says: run the formula $y = a + b_1 x_1 + b_2 x_2$

Using a formula in a regression

```r
res <- lm(y ~ x1 + x2, data=df)
summary(res) # show results
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4108 -0.9096  0.0794  0.9133  3.6410
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.7975     0.7526  -1.060   0.3041
## x1            1.3664     0.6185   2.209   0.0412 *
## x2            5.7549     0.3569  16.126 9.76e-12 ***
## ---
```

Formulas with interaction

If we changed `+` to `*`, we would add the interaction effect, ie we would run the formula
$y = a + b_1 x_1 + b_2 x_2 + b_{12} x_1 x_2$
Use `?formula` for more on this special syntax.

Formulas with interaction

```
res <- lm(y ~ x1 * x2, data=df)
summary(res) # show results
```

```
##
## Call:
## lm(formula = y ~ x1 * x2, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7761 -1.2702  0.0267  0.5813  4.2512
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.1305     0.7705  -1.467   0.1617
## x1            1.6573     0.6372   2.601   0.0193 *
## x2            6.5729     0.6819   9.639 4.57e-08 ***
## x1:x2        -0.9150     0.6564  -1.394   0.1824
```

Formulas with transformation

We could also use transformations. For example:

```r
res <- lm(y ~ x1 + log(x2), data=df)
```

```
## Warning in log(x2): NaNs produced
```

```r
summary(res) # show results
```

```
##
## Call:
## lm(formula = y ~ x1 + log(x2), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9336 -1.0740 -0.6565  0.3966  3.9343
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.05253    1.20643   5.846 0.000385 ***
## x1          -0.09403    1.07860  -0.087 0.932673
```

One-way analysis of variance (ANOVA)

Like t-test for multiple groups, again using a formula.

```
res = aov(height ~ gender * species, data=dplyr::starwars)
summary(res)
```

```
##                 Df Sum Sq Mean Sq F value   Pr(>F)
## gender           1   1779  1779.5   6.351   0.0162 *
## species         36  83802  2327.8   8.308 1.81e-09 ***
## gender:species   3    602   200.6   0.716   0.5488
## Residuals       37  10367   280.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
## 9 observations deleted due to missingness
```

Beyond Base R: the caret package

- k-nearest neighbours
- Linear regression
- Support vector machines
- Classification/regression trees
- Perceptrons
- Ensembles, including forests, bagging, boosting

https://topepo.github.io/caret

The `caret` package

The main Python competitor is `scikit-learn` which we will study later.

We won't go into detail on ML algorithms in this class.

Further reading

- https://www.statmethods.net/stats/ttest.html
- https://www.statmethods.net/stats/regression.html
- https://www.statmethods.net/stats/anova.html

Exercises

1. In the mpg dataset (part of the tidyverse), calculate the mean and standard deviation of the highway fuel efficiency.
2. Using group_by, calculate the mean and standard deviation of the highway fuel efficiency per manufacturer.
3. Calculate the correlation between highway fuel efficiency and engine size.
4. What was the average highway fuel efficiency in 1999 and in 2008?
5. Carry out a two-sample independent t-test between highway fuel efficiency in 1999 and 2008 and interpret the result.
6. Carry out a regression on highway fuel efficency by displacement.

Solution 1

library(tidyverse)

```
mean(mpg$hwy)
```

```
## [1] 23.44017
```

```
sd(mpg$hwy)
```

```
## [1] 5.954643
```

Solution 2

```
mpg %>% group_by(manufacturer) %>%
  summarise(mean=mean(hwy), sd=sd(hwy))
```

```
## # A tibble: 15 x 3
##    manufacturer  mean    sd
##    <chr>        <dbl> <dbl>
##  1 audi          26.4  2.18
##  2 chevrolet     21.9  5.11
##  3 dodge         17.9  3.57
##  4 ford          19.4  3.33
##  5 honda         32.6  2.55
##  6 hyundai       26.9  2.18
##  7 jeep          17.6  3.25
##  8 land rover    16.5  1.73
##  9 lincoln       17    1
## 10 mercury       18    1.15
## 11 nissan        24.6  5.09
```

Solution 3

```r
cor(mpg$hwy, mpg$displ)
```

```
## [1] -0.76602
```

Solution 4

```
mpg %>% group_by(year) %>%
  summarise(mean=mean(hwy), sd=sd(hwy))
```

```
## # A tibble: 2 x 3
##    year  mean    sd
##   <int> <dbl> <dbl>
## 1  1999  23.4  6.08
## 2  2008  23.5  5.85
```

Solution 5

```
mpg1999 <- mpg %>% filter(year == 1999)
mpg2008 <- mpg %>% filter(year == 2008)
t.test(mpg1999$hwy, mpg2008$hwy)
```

```
##
##  Welch Two Sample t-test
##
## data:  mpg1999$hwy and mpg2008$hwy
## t = -0.032864, df = 231.64, p-value = 0.9738
## alternative hypothesis: true difference in means is not equ
## 95 percent confidence interval:
##  -1.562854  1.511572
## sample estimates:
## mean of x mean of y
##  23.42735  23.45299
```

Solution 6

```
res = lm(hwy ~ displ, data=mpg)
summary(res)
```

```
##
## Call:
## lm(formula = hwy ~ displ, data = mpg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1039 -2.1646 -0.2242  2.0589 15.0105
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.6977     0.7204   49.55   <2e-16 ***
## displ        -3.5306     0.1945  -18.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```