Прикладной статистический анализ

Построение графиков

Перед переходом к статистическим темам рассмотрим несколько полезных функций для рисования графиков, которые окажутся полезными для иллюстрации материала. Начнем с того, что нарисуем параболу. Для этого воспользуемся функцией `plot()`, задав ей требуемый график поточечно:

```
x <- seq(0,1,0.01)
y <- x^2
plot(x, y)
```

Основной недостаток нашего результата состоит в том, что вместо графика мы получили набор круглешков для каждой точки. Ситуацию легко исправить указав параметр `type = "l"`, уточняющий что мы хотим соединить переданные точки линиями:

```
x <- seq(0,1,0.01)
y <- x^2
plot(x, y, type = "l")
```

Попробуйте также типы `"h"` и `"s"`:

```
x <- seq(0,1,0.01)
y <- x^2
plot(x, y, type = "l")
plot(x, y, type = "h")
plot(x, y, type = "s")
```

Как можно видеть, RStudio создало три отдельных полотна (вместе с удобной возможностью между ними переключаться). Если же вы хотите нарисовать несколько разных функций на одном полотне, то необходимо ограничеться одним вызовом `plot()` и дорисовывать оставшиеся функции через `lines()` и `points()`:

```
x <- seq(0,1,0.01)
y1 <- x^2
y2 <- sin(x*3)
y3 <- exp(-x)
plot(x, y1, type = "l")
points(x, y2)
lines(x, y3)
```

Результат можно сделать интереснее, добавив цвета (с помощью параметра `col`) и названия для осей:

```
x <- seq(0,1,0.01)
y1 <- x^2
y2 <- sin(x*3)
y3 <- exp(-x)
plot(x, y1, type = "l", col = "red", xlab = "Stuff concentration", ylab = "Something reaction")
points(x, y2, col = "green")
lines(x, y3, col = "blue")
```

Наконец, если вы хотите нарисовать графики в нужном вам масштабе, то можете указать требуемые участки осей (параметры `xlim` и `ylim`) и отношение сторон полотна (`y/x`, параметр `asp`):

```
x <- seq(0,1,0.01)
y1 <- x^2
y2 <- sin(x*3)
y3 <- exp(-x)
plot(x, y1, type = "l", col = "red", xlab = "Stuff concentration", ylab = "Something reaction",
points(x, y2, col = "green")
lines(x, y3, col = "blue")
```

Рисование полигонов и подграфиков

Часто очень удобно бывает продемонстрировать что-либо с помощью закрашенного подграфика. Этого эффекта можно добиться с помощью функции `polygon`, которая рисует полигон, заданный набором точек:

```
# `polygon()` лишь дорисовывает на полотне, не создавая свое,
# поэтому его сначала необходимо создать
# `plot.new()` позволяет это быстро сделать, не рисуя других графиков.
plot.new()
# (0, 0), (1, 0), (0.5, 1)
triangleX <- c(0, 1, 0.5)
triangleY <- c(0, 0, 1)
polygon(triangleX, triangleY)

rectX <- c(0.05, 0.05, 0.7, 0.7)
rectY <- c(0.2, 0.65, 0.65, 0.2)
polygon(rectX, rectY)
```

Полигон можно закрашивать с помощью параметра `col` и убирать у него гранцу (черную обводку) установив `border = NA`:

```
plot.new()
# (0, 0), (1, 0), (0.5, 1)
triangleX <- c(0, 1, 0.5)
triangleY <- c(0, 0, 1)
polygon(triangleX, triangleY, col = "red")

rectX <- c(0.05, 0.05, 0.7, 0.7)
rectY <- c(0.2, 0.65, 0.65, 0.2)
polygon(rectX, rectY, col = "green", border = NA)
```

Попробуем применить полигоны для закрашивания подграфиков:

```
x <- seq(0,1,0.01)
y <- sin(x*3)
plot(x, y, type = "l", col = "red")
polygon(x, y, col = "red", border = NA)
```

При таком подходе закрашенная область получется немного скривленной. Это можно исправить добавив ещё две точки, соединяющие рисуемый полигон с осью абсцисс:

```
x <- seq(0,1,0.001)
y <- sin(x*3)
plot(x, y, type = "l", col = "red")
polygon(c(x, x[length(x)], x[1]), c(y, 0, 0), col = "red", border = NA)
```

Мы также можем добавить графикам прозрачности, добавив цвета альфа-канал с помощью команды `adjustcolor()`. Например: `adjustcolor("red", alpha.f = .25)`. Это позволит нам также показывать пересечение двух подграфиков:

```
x <- seq(0,1,0.001)
y1 <- sin(x*3)
y2 <- sin(1.5 + x*3)
plot(x, y1, type = "l", col = "red")
lines(x, y2, col = "blue")
polygon(c(x, x[length(x)], x[1]), c(y1, 0, 0), col = adjustcolor("red", alpha.f = .25), border =
polygon(c(x, x[length(x)], x[1]), c(y2, 0, 0), col = adjustcolor("blue", alpha.f = .25), border
```

Упражнение 1

Создайте функцию `normPlots()`, принимающую на вход четыре параметра `mean1`, `mean2`, `sd`, `level` и рисующую две плотности нормального распределения с средними значениями `mean1` (крастным цветом), `mean2` (синим) и дисперсией `sd^2`. Плостности должны иметь закрашенные подграфики. Кроме того, если `level != NA`, она должна рисовать вертикальную черту, выходящую из точки `(0, level)` и заканчивающуюся на вершине графика (на уровне пиков плотностей).
Плотность нормального распределения вычисляется функцией `dnorm()`.

Проверка гипотез

Мы разработали новый революционный препарат, позволяющий людям повысить своё IQ, скажем, на двадцать пунктов и теперь хотим доказать его работоспособность. Для этого мы вводим две гипотезы: H_0 - наше лекарство ничего не делает и H_a - оно работает как обещано. Для проверки этой гипотезы мы проводим простой тест - выбираем случайного человека с улицы (только одного, препарат дорогой!), скармливаем ему лекарство и даём пройти IQ-тест. Если в результате у него будет больше 120IQ, то считаем что наше лекарство работает (принимаем альтернативную гипотезу H_a) или же меньше (отвергаем альтернативную гипотезу и принимаем H_0).

Картинка это иллюстрирующая, и предполагающая нормальное распределение IQ, может быть получена так (красный график соответствует основной гипотезе, а синий - альтернативной):

```
normPlots(100, 120, 10, level = 120)
```

Как видно из картинки, даже если наш препарат эффективен, - вероятность тесту объявить наш препарат нерабочим равна $\alpha_1 = 50$ (ошибка второго рода). В то же время вероятность α_2 неоправданно решить, что препарат эффективен, когда это не так соответствует лишь небольшому кусочку красной плотности (ошибка второго рода α_2).

Двигая пограничное значение 120IQ влево или вправо мы может изменять значения этих ошибок, но среди них нет строго оптимального: уменьшение ошибки первого рода приводит к росту ошибки второго и наоборот.

На практике обычно фиксируют ошибку первого рода равной $\alpha_1 = 0.05$. Вместо же ошибки второго рода, обычно говорят о **мощности** теста $1 - \alpha_2$, описывающую вероятность действительно принять альтернативную гипотезу, если она верна. Повышения мощности добиваются увеличением выборки.

В нашем случае, например, будет необходимо тестировать препарат на большем количестве людей. Действительно, если мы протестируем 100 людей и рассмотрим их среднее значение IQ, то оно также окажется норамльно распределено с тем же средним, но в 10 раз меньшей дисперсией. Тогда тест с пограничным значением 110 должен дать очень низкие ошибки первого и второго рода, а соответственно и высокую мощность:

```
normPlots(100, 120, 1, level = 110)
```

Обычно тесты проектируют так, чтобы они имели мощность не менее 0.8, а лучше 0.9 или 0.95.

Размер эффекта

Такой замечательный тест на всего 100 людях у нас получился благодаря тому, что наш препарат обещал быть слишком эффективным. Если же мы возбмём более скромный прирос в 2IQ, то результаты будут скромнее:

```
normPlots(100, 102, 10, level = 101)
normPlots(100, 102, 1, level = 101)
```

Такое изменение было вызвано уменьшением ожидаемого эффекта нашего препарата. Наоборот, достаточно увеличив этот эффект (пусть препарат добавляет 100IQ) мы можем ограничиться и одним испытанием:

```
normPlots(100, 200, 10, level = 150)
```

Отсюда можно вынести, что больший размер эффекта влечет к большей мощности. Если же размер эффекта близок к нулю, то чтобы добиться достаточной мощности нам может понадобиться очень большое число испытаний:

```
normPlots(100, 100.1, 1, level = 150)
```

Заметим также, что размер эффекта задается не только разницей средних значений, но и дисперсией распределения (растянотостью графика). Для его определения обычно используется (коэффициент d Коэна)[https://en.wikipedia.org/wiki/Effect_size]
с объединенным стандартным отклонением (pooled standard deviation):

$$d = \frac{\bar{x}_1 - \bar{x}_2}{s},$$

$$s = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}},$$

где $s_i^2 = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (x_{j,i} - \bar{x}_1)^2$.

Обычно 0.2 считается маленьким размером эффекта, 0.5 средним, а 0.8 уже большим.

Упражнение 2

Какие значения коэффициента вы бы ожидали увидеть для выборок в случаях, когда препарат увеличивает IQ на 20 пунктов? На 2?

Напишите функцию для вычисления Коэффициент d Коэна и посчитайте размер эффекта для симулированных данных в случае приростов 20 и 2 IQ.
Сравните их значения.

Тест Стьюдетна

t-тест Стьюдента это один из наиболее распространенных тестов в статистике. Он используется для определения равенства средних значений двух разных групп (размеров n_1 и n_2). Предполагается, что обе группы имеют нормальное распределение с одинаковыми дисперсиями. За основную гипотезу берут предположение о равенстве средних значений, а за альтернативную - наоборот, что они различаются (хоть на какую величину).

Например, мы можем измерять IQ двух групп - одной мы не даём никаких препаратов, другим даём повышающий IQ. Затем как раз проверяем гипотезу о том, что IQ изменилось против того, что препарат не оказал никакого эффекта.

Если основная гипотеза верна, то мы можем вычислить t-статистику, имющую распределение Стьюдента с $n_1 + n_2 - 2$ степенями свободы. Перед тем, как двигаться дальше - найдем простой способ подсчета t-статистики.

Для проведения этого теста в R имеется функция `t.test()`. Давайте попробуем применить её на каких-нибудь случайно сгенерированных данных:

```
x <- rnorm(1000)
y <- rnorm(1000)
t.test(x,y)
```

Как вытащить из этого результата требуемую t-статистику? Или какое-нибудь другое из приведенных значений? Общий метод разрешения подобных ситуаций - использовать функцию `names()`:

```
ttest <- t.test(x,y)
names(ttest)
```

Интересующее нас значение t-статистики записано в значении `statistic`:

```
ttest$statistic
ttest[["statistic"]]
```

Определение размера выборки необходимого для теста Стьюдетна

Выше мы упоминали такие величины как ошибка первого рода, мощность, размер эффекта и размер выборки, а также связи между ними. Задав все кроме одной из них, мы можем получать оставшуюся величину. В основном нас будет интересовать определение необходимого размера выборки для проведения достаточно мощного теста, так как вероятность ошибки первого рода более или менее фиксирована, а на величину эффекта у нас часто нет никакого влияния. В R для этого можно использовать функцию `power.t.test()`, основными её параметрами которой являются:

параметер	что он описывает
n	размер выборки (для каждой группы)
delta, sd	размер эффекта: разница между средними значениями в двух гипотезах и дисперсия их распределений
sig.level	вероятность ошибки первого рода, обычно = 0.05
power	мощность, обычно не менее 0.8

Требуется указать ей все эти параметры кроме одного (по-умолчанию указаны `sd = 1` и `sig.level = 0.05`, выставьте их в NULL, если именно их требуется найти). Посмотрим какая нам требовалась выборка, для того, чтобы удостовериться в работостособности нашего препарата для увеличения IQ с мощностью в 0.8:

```
power.t.test(delta = 2, sd = 10, power = 0.8)$n
```

Если же нам потребуется гарантировать большую мощность теста, то потребуется обеспечить больший размер выборки:

```
print(paste("For 0.8:", power.t.test(delta = 20, sd = 10, power = 0.8)$n))
print(paste("For 0.85:", power.t.test(delta = 20, sd = 10, power = 0.85)$n))
print(paste("For 0.9:", power.t.test(delta = 20, sd = 10, power = 0.9)$n))
print(paste("For 0.95:", power.t.test(delta = 20, sd = 10, power = 0.95)$n))
```

Такие маленькие размеры выборок требуются из-за нашего громадного размера эффекта. Сравните результаты выше с ситуацией, когда наш препарат увеличивает IQ лишь на 2 пункта:

```
print(paste("For 0.8:", power.t.test(delta = 2, sd = 10, power = 0.8)$n))
print(paste("For 0.85:", power.t.test(delta = 2, sd = 10, power = 0.85)$n))
print(paste("For 0.9:", power.t.test(delta = 2, sd = 10, power = 0.9)$n))
print(paste("For 0.95:", power.t.test(delta = 2, sd = 10, power = 0.95)$n))
```

Определение разницы между средними значениями двух выборок

Что если мы не уверены на сколько именно наш препарат увеличивает IQ? Прикинуть это можно с помощью точечной оценки $\bar{x}_1 - \bar{x}_2$. Но насколько точна эта оценка? Это можно оценить с помощью доверительных интервалов. Функция `t.test()` кроме прочего находит и доверительный интервал интервал уровня 0.95, в котором, по нашим данным, лежит разница средних значений:

```
x <- rnorm(10, mean = 120, sd = 10)
y <- rnorm(10, mean = 100, sd = 10)
ttest <- t.test(x, y)
ttest$conf.int
```

Отметим что размер интервала уменьшается как корень из n вместе с ростом размера выборки.

Упражнение 3

Напишите функцию, находящую размер интервала по данному размеру выборки, двух средних значений и среднеквадратичного отклонения.

С её помощью проведите симуляции для каких-нибудь значений средних значений и среднеквадратичного отклонения. Нарисуйте график зависимости размера интервала от размера выборки. Можете использовать для этого функцию
`replicate(<amount_of_repetitions>, <operation>)`

Упражнение 4

Иногда контрольную группу полагают больше группы лечения. Такое случается, например,в медицинских исследованиях, когда обслуживание одного пациента из группы лечения (группы с новым лекарством) значительно дороже обслуживания пациента из контрольной группы (группы со старым лекарством).

Но функция `power.t.test()` может вычислять мощность лишь при одинаковом размере групп. Напишите собственную функцию вычисления мощности
`simulatePower(n1, n2, delta, sd = 1, sig.level = 0.05, steps = 10^4)` , основанную на симуляциях.

Проверьте её:

```
print("For the sample size of 10:")
simulatePower(10, 10, 2, sd = 10)
power.t.test(delta = 2, sd = 10, n = 10)$power
print("For the sample size of 20:")
simulatePower(20, 20, 2, sd = 10)
power.t.test(delta = 2, sd = 10, n = 20)$power
print("For the sample size of 100:")
simulatePower(100, 100, 2, sd = 10)
power.t.test(delta = 2, sd = 10, n = 100)$power
```

Упражнение 5

Нарисуйте на одном полотне четыре графика зависимости мощности от размера выборки второй группы n при размерах первой (контрольной) n, $2n$, $3n$ и $4n$. Полагайте средние значения равными 100 и 102, а среднеквадратичное отклонение - 10. Рекомендуется проводить симуляции для значений n от 10 до 1000 (не обязательно всех, достаточно для каждых 100) и на каждую симуляцию использовать порядка 1000 шагов.

Сделайте выводы.

> Напомним, что чтобы определять была ли принята гипотеза достаточно проверить, что полученное p-value оказалось меньше 0.5.

```
sampleSizes <- seq(10, 1000, 10)
single <- c()
double <- c()
triple <- c()
quadruple <- c()
for(i in sampleSizes) {
  single <- c(single, simulatePower(i, i, 2, sd = 10, steps=10^3))
  double <- c(double, simulatePower(2 * i, i, 2, sd = 10, steps=10^3))
  triple <- c(triple, simulatePower(3 * i, i, 2, sd = 10, steps=10^3))
  quadruple <- c(quadruple, simulatePower(4 * i, i, 2, sd = 10, steps=10^3))
}
plot(sampleSizes, single, type = "l", col = "purple")
lines(sampleSizes, double, col = "red")
lines(sampleSizes, triple, col = "green")
lines(sampleSizes, quadruple, col = "blue")
```