R Users Guide to Stat 201: Chapter 11

Michael Shyne, 2017

Chapter 11: Goodness-of-Fit and Contingency Tables

Chapter 11 covers goodness-of-fit tests for one dimensional frequency tables and tests for independence for two dimensional contingency tables (which are also frequency tables). Both tests use the χ^2 (chi-squared) distribution and thus are handled similarly in R.

Goodness-of-fit tests

A goodness-of-fit test will test whether a one dimensional table of frequency counts matches a given frequency distribution. We will use the `chisq.test()` function.

Often we are given data already in table form, which is what the `chisq.test()` function expects. For example, suppose with are given the following results of a six-sided die rolled 100 times.

Result	One	Two	Three	Four	Five	Six
Count	15	28	17	10	15	15

We can conduct a test of whether the die is "fair", in other words whether the frequencies match a uniform distribution.

```
dice <- c(15, 28, 17, 10, 15, 15)

# Is the die "fair"?
chisq.test(dice)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  dice
## X-squared = 10.88, df = 5, p-value = 0.05381
```

The p-value of 0.05381 compels us, at a significance level of $\alpha = 0.05$, to not reject the null hypothesis the sample of die rolls does come from a uniform distribution. However, the p-value is worryingly close to the 0.05 level, which might indicate that more tests are warranted.

If we are working with data that has not been summarized into a frequency table, as we have seen before, we can create a frequency table of categorical data with the `table()` function. Suppose we want to test whether the number of cylinders of the cars in the `mtcars` data set are uniformly distributed. Though the `cyl` variable is numeric, it actually represents a ordinal variable and we can use it as such.

```
# Build frequency table for number of cylinders
cyl.table <- table(mtcars$cyl)

# Are the number of cylinders uniformly distributed?
chisq.test(cyl.table)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  cyl.table
```

```
## X-squared = 2.3125, df = 2, p-value = 0.3147
```

In order to test a frequency table against a given non-uniform distribution, we will include `p=` to the function call with a vector of proportions. The proportion vector must be the same length as the frequency table and the proportions must add to one. Suppose we expect 2/5 of cars to have 4 cylinders, 2/5 to have 6 and the remaining 1/5 to have 8 cylinders.

```
cyl.prop <- c(0.4, 0.4, 0.2)

# Do the cars in data set match expected proportions?
chisq.test(cyl.table, p=cyl.prop)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  cyl.table
## X-squared = 11.906, df = 2, p-value = 0.002598
```

Sometimes, we have expected ratios between category values rather than expected proportions. We can easily calculate proportions by dividing each ratio value by the sum of all ration values. For example, suppose we designed the die used to generate the roll data to have twice as many 2's as other numbers.

```
# We expect twice as many 2's
dice.ratio <- c(1, 2, 1, 1, 1, 1)

# Expected ratios as proportions
dice.prop <- dice.ratio / sum(dice.ratio)
dice.prop
```

```
## [1] 0.1428571 0.2857143 0.1428571 0.1428571 0.1428571 0.1428571
```

```
# Do the die rolls match expect distribution?
chisq.test(dice, p=dice.prop)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  dice
## X-squared = 1.92, df = 5, p-value = 0.8601
```

Contincency tables

A test for independence tests whether frequency tables of two dimensions (rows and columns) or contingency tables demonstrate an association between the variables or factors represented. Like with goodness-of-fit tests, data is often presented in summarized form. In R, this means data in a table or matrix.

Suppose we have data from a drug screening process at a large company. For each subject in the sample, they either tested positive for drugs or they did not and, after a more expensive but known reliable test, they either had taken drugs or they did not. The data is summarized in the table below.

	Screening Results	
Taken Drugs?	Positive	Negative
Yes	15	12
No	84	98

To use this data in R, we will use the `matrix()` function, which takes a vector of values and arranges them in a two dimensional matrix. When defining the matrix, we can specify the number of rows or the number of columns with the `nrow=` or `ncol=` parameters, respectively. Only one needs to be used. Also, we can specify whether the values of the vector are added to the matrix by rows or by columns. The default method is by column, but we can add `byrow=TRUE` to add by row. It is useful to experiment with some simple examples of the `matrix()` function so you are clear on how to get desired results before moving to more complicated data.

```
# Specify number of cols, add values by row
drug.matrix <- matrix(c(15,12,84,98), ncol = 2, byrow = TRUE,
                      dimnames = list(c("Yes", "No"),     # Adding dimension labels
                                      c("Pos", "Neg")))  # for readability

drug.matrix
```

```
##     Pos Neg
## Yes  15  12
## No   84  98
```

Once we have the data in the proper form, the test for independence is conducted by passing the matrix or two dimensional table to `chisq.test()`. This function, like `prop.test()` previously, by default performs a continuity correction. This wasn't an issue when we were conducting one dimensional goodness-of-fit tests, but it can affect the results of tests on two dimensional data. In order to be consistent with StatCrunch, we will want to disallow the correction.

```
# Are drug screening results independent of drug use?
chisq.test(drug.matrix, correct = FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  drug.matrix
## X-squared = 0.83362, df = 1, p-value = 0.3612
```

With a p-value of 0.3612, we would fail to reject the null hypothesis of independence. In other words, we don't have evidence that the drug screening results are affected by whether the subject actually uses drugs or not.

High dimension tables and apply

Sometimes we are given data in a high dimension table. Consider the built-in `Titanic` data set.

```
str(Titanic)
```

```
##  table [1:4, 1:2, 1:2, 1:2] 0 0 35 0 0 0 17 0 118 154 ...
##  - attr(*, "dimnames")=List of 4
##   ..$ Class   : chr [1:4] "1st" "2nd" "3rd" "Crew"
##   ..$ Sex     : chr [1:2] "Male" "Female"
##   ..$ Age     : chr [1:2] "Child" "Adult"
##   ..$ Survived: chr [1:2] "No" "Yes"
```

We can see from the structure that this is a four dimensional table, the dimensions being `Class`, `Sex`, `Age` and `Survived`. In order to perform a test of independence, using known techniques, we will have to reduce the data to two dimensions. Suppose we are interested in whether gender had an effect on survival. We will use the `apply()` function to traverse specified dimensions and apply a function, in this case `sum()`. We can see from the `str()` output, sex is dimension 2 and survival is dimension 4.

```r
# Reduce the dimension of the Titanic data set
titantic.sex.survive <- apply(Titanic, c(2,4), sum)
titantic.sex.survive
```

```
##         Survived
## Sex        No Yes
##   Male   1364 367
##   Female  126 344
```

```r
# Did sex affect survival?
chisq.test(titantic.sex.survive, correct = FALSE)
```

```
##
##  Pearson's Chi-squared test
##
## data:  titantic.sex.survive
## X-squared = 456.87, df = 1, p-value < 2.2e-16
```

From the low p-value, we can conclude that there is an association between sex and survival among the Titanic passengers.

I am not going into a full discussion of the `apply()` function or related functions such as `sapply()` and `tapply()`. I encourage you to investigate them on your own. They can be invaluable tools in the right situation.

Tests for independence with unsummarized data

Of course, many times we need to work with data that is not summarized. We can use the `table()` function to create two or higher dimension tables. Let's return to the `mtcars` data set. Suppose we are interested in whether number of cylinders has a association with fuel efficiency. The `mpg` variable is a numeric variable, so we will need to convert it into a categorical variable. We will define a new variable `fuel.eff` which will take the values "High" if `mpg` is greater than the mean mpg and "Low" otherwise. Then we will summarize the factors we are interested in into a two dimensional table.

```r
mpg.mean <- mean(mtcars$mpg)

# Define new variable with values "High" or "Low"
mtcars$fuel.eff[mtcars$mpg > mpg.mean] <- "High"
mtcars$fuel.eff[mtcars$mpg <= mpg.mean] <- "Low"

# Build a table
cars.table <- table(mtcars$cyl, mtcars$fuel.eff)
cars.table
```

```
##
##     High Low
##   4   11   0
##   6    3   4
##   8    0  14
```

```r
# Is there an association between fuel efficiency and number of cylinders
chisq.test(cars.table, correct = FALSE)
```

```
## Warning in chisq.test(cars.table, correct = FALSE): Chi-squared
## approximation may be incorrect
```

```
##
```

```
##  Pearson's Chi-squared test
##
## data:  cars.table
## X-squared = 25.034, df = 2, p-value = 3.664e-06
```

Notice that the `chisq.test()` function warned us that the χ^2 might be incorrect. This is because we have cells with low counts (a couple of them are zero). However, the test gave strong evidence of an association, so we can probably accept the result.

License