FASS512: Correlations, linear regression. Tests for categorical variables.

Professor Patrick Rebuschat, [p.rebuschat@lancaster.ac.uk](mailto:p.rebuschat@lancaster.ac.uk)

In the first part of this handout, we will look at **how to analyse relationships between two continuous data variables**. We will consider two standard techniques: correlation and simple linear regression.

By the end of this session, you should:

* Understand what a correlation is
* How to interpret the value of a correlation coefficient
* What the different kinds of correlation coefficient are
* How to run a correlation test in R
* How to express the linear relationship between two variables in the form of a model equation (simple linear regression)
* The meaning of the terms: predictor variable, outcome variable, intercept, and slope
* How to run a simple linear regression using R
* How you can use a model equation to predict new data values

In the second part, we will then look at **how to perform and interpret tests for categorical variables**, also referred to as ‘count data.’

We will see how to set up tables of count data and how to run three different tests (Pearson’s chi-squared test, Fisher’s exact text, and the likelihood-ratio test). We will also consider how to measure an overall effect size, and we will see how we can identify which counts are contributing most to an overall effect.

By the end of this session you should:

* Know what a contingency table is, including the difference between ordered and unordered contingency tables
* Know how to create a contingency table in R
* Appreciate the main considerations underlying the choice between Pearson’s chi-squared test, Fisher’s exact test, and the likelihood-ratio test
* Know how to perform and interpret these three tests using R
* Know how to calculate and interpret three measures of effect size for contingency tables (the Odds Ratio, phi, and Cramer’s V)
* Know how to access and interpret the residuals from a chi-squared test

**References for this handout**

Many of the examples and data files from our class come from these excellent textbooks:

* Andrews, M. (2021). *Doing data science in R*. Sage.
* Brown, D. S. (2021). *Statistics and data visualization using R. The art and practice of data analysis*. Sage.
* Cumming, G. & Calin-Jaegeman, R. J. (2017). *Introduction to the New Statistics: Estimation, Open Science, and Beyond*. Routledge.
* Crawley, M. J. (2013). *The R book*. Wiley.
* Fogarty, B. J. (2019). *Quantitative social science data with R*. Sage.
* Winter, B. (2019). *Statistics for linguists. An introduction using R*. Routledge.

References (not intended as a recommended reading list)

Camilli, G. (1990). The test of homogeneity for 2 x 2 contingency tables: A review of and some personal opinions on the controversy. *Psychological Bulletin, 108*, 135­145.

Cochran, W. G. (1954). Some methods for strengthening the common χ2 tests. *Biometrics, 10*, 417­451.

Cohen, J. (1992). A power primer. *Psychological Bulletin, 112,* 155-159.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics, 19*, 61­74.

Gries, S. Th. (2005). Null­hypothesis significance testing of word frequencies: A follow­up on Kilgarriff. *Corpus Linguistics and Linguistic Theory, 1­2*, 277­294.

Lock, R. H. (1993) 1993 New car data. *Journal of Statistics Education* 1(1).

Are you ready? Then let’s start on the next page! ☟

Task 1: *Loading and inspecting our data set*

The example data set that we will be using in this session is called Cars93, and it is contained in the add-on package MASS. MASS is now automatically included in distributions of R, so you shouldn’t have any problems accessing it. All you need to do is to type the following command.

library(MASS)

The built-in data set Cars93 can then be loaded with the following command.

data(Cars93)

This data set, collected in 1993, contains various pieces of information about 93 kinds of cars for sale in the United States. This data is frequently used to illustrate R analyses; it is fairly easy to understand and doesn’t require any specialist background.

For each car, you will find information on manufacturer, model, type (e.g., small, compact, large), price (minimum and maximum, in $1,000), fuel economy (measured in MPG, miles per gallon), engine size (in liters), horsepower (the power of the engine), country of origin, and so forth.

For more information about the data, you can type:

help(Cars93)

Task 2: *Correlations: Visual inspection*

*The relationship between price and horsepower*

When it comes to cars, we might hypothesize that more powerful cars are likely to cost more than less powerful ones. The Cars93 data set contains two variables that can help us to examine how far this plausible relationship holds true: Max.Price, which is the maximum price at which a particular type of car retailed at the time, and Horsepower, which is a measure of the power of the car.

A few weeks ago, we looked at scatterplots and saw how they could be used to visualize relationships between two continuous variables.

So, we could make a useful start on our problem by plotting the values of these two variables against each other in a scatterplot:

plot(

Cars93$Horsepower,

Cars93$Max.Price,

pch = 18,

cex = 2,

col = "turquoise",

xlab = "Horsepower",

ylab = "Max. Price",

main = "Horsepower against Max. Price for 93 Cars in 1993"

)

As you might remember, in the plot() function, the first two elements inside the brackets are the variables that we want to plot against each other. The arguments pch, cex, and col define the shape, the size and color of our markers. And xlab, ylab, and main refer to the X and Y axis labels as well as the chart title. (The R graph gallery is a helpful resource and looks great… You can click [here](https://r-graph-gallery.com/scatterplot.html) to check out basics about scatterplots.)



If we look at the shape of this cloud of data points, it seems that there exists a positive relationship between horsepower and maximum price: most of the cloud follows a fairly narrow diagonal band from bottom left to mid/top right, indicating that, as horsepower values get bigger, so do maximum prices. It would therefore seem to make sense to take the analysis a step further.

Task 3: *Correlations: Statistical tests*

If we want to examine the degree of association between two continuous variables, we can use a measure called a correlation.[[1]](#footnote-2) This tells us, on a fixed scale of -1 to +1, how closely associated the two variables are.

A **positive correlation** tells us that the two variables go up in step with each other, as in the example in the graph above.

A **negative (or inverse) correlation** tells us that, as one variable goes up, the other one goes down.

As a general rule of thumb, we could say:

* a correlation of **0 to 0.1** (or 0 to -0.1) tells us that there is pretty much **no relationship** at all;
* a correlation of between **0.1 and 0.3** (or 0 and -0.3) shows a **small** relationship;
* a correlation of between **0.3 and 0.5** (or -0.3 and -0.5) is a **moderate** relationship;
* a correlation of between **0.5 and 1.0** (or -0.5 and -1.0) is a **large** relationship; and
* a correlation coefficient of **±1** shows a **perfect** correlation.

These interpretations are based on Cohen’s (1992, p. 157) suggestions about effect sizes; it is the same scale that we use for interpreting Phi and Cramer’s V below.

Depending on what books and articles you read, you will sometimes see slightly different ranges and interpretations suggested. For example, the following break-down is also quite common.

* -0.7 to -1 strong negative correlation
* -0.3 to -0.7 moderate negative correlation
* -0.1 to -0.3 weak negative correlation
* 0 no correlation whatsoever
* 0.1 to 0.3 weak positive correlation
* 0.3 to 0.7 moderate positive correlation
* 0.7 to 1 strong positive correlation

When interpreting the relative importance of correlation coefficients, we need to use our common sense and what we already know about our data and research questions.

For instance, if we were correlating the results of two achievement tests that were intended to assess **exactly the same** construct at exactly the same level, we might not be too impressed by a supposedly ‘strong’ relationship of just 0.5! We might want to see at least 0.7, or even something higher than that. However, in a different research context, 0.5 might indeed be considered a strong relationship.

We also need to bear in mind that **correlations merely show that a relationship exists between two things**. They do not, by themselves, demonstrate that one variable causes an increase or decrease in the other.

*Different correlation tests*

In most statistical software, including R, we normally have a choice between three types of correlation test.

* Pearson’s r
* Spearman’s rho (ρ)
* Kendall’s tau (τ)

Pearson correlations are only really appropriate when the data are continuous and approximately normally distributed. If they are not, and you do not employ some kind of transformation in order to bring them closer to a normal distribution, then you will probably need to work with either Spearman or Kendall correlations, which are based on the rank order of the data values rather than on the data values themselves (in a similar way to the Wilcoxon-Mann-Whitney test).[[2]](#footnote-3) Both are widely used, though Spearman is perhaps more commonly encountered. The two rank-order correlations are also valuable when the data are otherwise approximately normal but contain one or two extreme values (or ‘outliers’), as the Pearson coefficient is quite sensitive to bias by these.

Most of the time, the different correlation tests tend to give similar results, although Kendall correlations typically show somewhat smaller sizes.

To illustrate this, here are the results of three correlation tests performed using R on exactly the same set of data:

* Pearson r: 0.677
* Spearman: 0.604
* Kendall: 0.436

*Performing correlation tests in R*

To perform a correlation test in R, we use the cor.test() function as in the following example.

cor.test(Cars93$Horsepower,

Cars93$Max.Price,

method = "spearman",

exact = FALSE)

Here, Cars93$Horsepower and Cars93$Max.Price are the two variables of interest, and the statement method = "spearman" tells R to compute the Spearman rank correlation. For a Kendall correlation, we would simply replace "spearman" with "kendall". For a Pearson correlation, we can either substitute "pearson" or omit this statement entirely, as Pearson is the default setting.

The output for the Spearman rank correlation looks like the following.

Spearman’s rank correlation rho

data: Cars93$Horsepower and Cars93$Max.Price

S = 23917, p-value < 2.2e-16

alternative hypothesis: true rho is not equal to 0

sample estimates:

rho

0.8215706

Here, the main item of interest is the correlation coefficient, shown underneath "rho" and highlighted in red. You will see it is a large correlation by most definitions: 0.82. And the p-value tells us that it is statistically significantly different from zero at p < 0.01.

For a Pearson correlation, R also automatically provides a 95% confidence interval, highlighted in the output below in blue, for the correlation coefficient r.

cor.test(Cars93$Horsepower, Cars93$Max.Price, method = "pearson")

Pearson’s product-moment correlation

data: Cars93$Horsepower and Cars93$Max.Price

t = 10.636, df = 91, p-value < 2.2e-16

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.6373975 0.8232999

sample estimates:

cor

0.7444447

R does not do this for a Spearman or Kendall correlation.

cor.test(Cars93$Horsepower, Cars93$Max.Price, method = "kendall")

Kendall’s rank correlation tau

data: Cars93$Horsepower and Cars93$Max.Price

z = 8.9562, p-value < 2.2e-16

alternative hypothesis: true tau is not equal to 0

sample estimates:

tau

0.6380523

However, for a Spearman correlation coefficient, a calculation can be done semi-manually as follows.

rho <- add here the actual rho value

n <- add here number of pairs of observations

z <- 0.5 \* log((1 + rho)/(1 - rho))

lci <- z - (1.96/sqrt(n - 3))

uci <- z + (1.96/sqrt(n - 3))

c((exp(2\*lci)-1)/(exp(2\*lci)+1),(exp(2\*uci)-1)/(exp(2\*uci)+1))

In the first line of this code, just add the actual value of rho in the first line rho <-. In the second line, add the number of pairs of observations after n <-. Then simply copy and paste the remaining lines into R.

So, for our Cars93 data, we enter 0.8215706 as the correlation coefficient (rho) and 93 as the number of pairs of data values.

rho <- 0.8215706

n <- 93

z <- 0.5 \* log((1 + rho)/(1 - rho))

lci <- z - (1.96/sqrt(n - 3))

uci <- z + (1.96/sqrt(n - 3))

c((exp(2\*lci)-1)/(exp(2\*lci)+1),(exp(2\*uci)-1)/(exp(2\*uci)+1))

[1] 0.7420511 0.8782888

The resulting 95% CI for an estimated population rho of 0.821 is 0.742 to 0.878, which is a relatively narrow CI and confirms that we are dealing here with a genuinely large effect size. Both the estimate, and the lower bound of the CI, are well above Cohen’s 0.5 lower limit for a large effect size, and they are also above the alternative 0.7 lower limit sometimes suggested in textbooks.

*How to report the results*

To report the results of our correlation tests, we need to report the test statistic, the degrees of freedom and the p-value. For example, to report the Pearson correlation test, we could write the following.

“Our analysis revealed a strong positive relationship between horsepower and max. price, *r*(91) = .75, *p* < .001, 95% CI [.64, .82].”

Note that the r-value should be rounded to two decimal places. The degrees of freedom are N – 2. Since there are 93 cars, the degrees of freedom equal 91. Finally, remember that normally we should report the exact p-values. However, when the values are below .01, we should report them as inequalities, either p < .01 or p < .001. The actual p-value is 0.00000000000000022, hence p < .001 in our report.

Task 4: *Simple Linear Regression*

A correlation is merely a rough measure of the existence and magnitude of a relationship between two variables. It does not enable us to make actual predictions about other data values that have not been observed in our sample, or to characterize the relationship between the variables more explicitly in the form of a theoretical model equation. If we want to do that, we need to use another technique called (linear) **regression**.

Regression is a statistical method that is broadly equivalent to drawing a ‘best-fit’ straight line through the cloud of points on the plot.

A linear regression line is characterized by an equation of the type:

*y = a + bx*

Here, *y* is the **dependent (or outcome) variable** and *x* is the **independent (or predictor)** variable.

In our example case of cars, we are assuming that the power of the car is a predictor of its maximum price, so Horsepower is the predictor variable (x) and Max.Price is the outcome variable (y).

On a plot, we would normally plot the predictor variable along the horizontal (x) axis and the outcome variable along the vertical (y) axis.

In the equation above, the values *a* and *b* are known as the **coefficients** of the regression equation.

The coefficient *a* is called the **intercept** and is the point at which the regression line hits the vertical (y) axis, when x = 0. Below, the intercept data will be highlighted in blue.

The coefficient *b* is known as the **slope** of the regression line and determines how steep its gradient is. This will be highlighted in green.

Both of these values are estimated automatically by the R software on the basis of the data.

To fit a simple linear regression in R, we use the lm() function, as in the example below. lm stands for ‘linear model.’

our\_model <- lm(Max.Price ~ Horsepower, data = Cars93)

Here, our\_model is merely a random name that we choose to give to the regression model, so that we can refer to it later on in our R session.

The first argument in the lm() function is the formula, here Max.Price ~ Horsepower. The outcome variable (Max.Price) is placed to the left of the tilde (the squiggly line), and the predictor variable (Horsepower) is placed on the right. The name of our data table within R comes after the data argument, hence data = Cars93.

To examine the output of the model, we can type the following command.

our\_model

Call:

lm(formula = Max.Price ~ Horsepower, data = Cars93)

Coefficients:

(Intercept) Horsepower

-0.6513 0.1568

For a more comprehensive output, we can use the summary() function as follows.

summary(our\_model)

Call:

lm(formula = Max.Price ~ Horsepower, data = Cars93)

Residuals:

Min 1Q Median 3Q Max

-13.284 -3.795 -1.051 2.727 46.629

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.65126 2.25493 -0.289 0.773

Horsepower 0.15679 0.01474 10.636 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ‘ 1

Residual standard error: 7.405 on 91 degrees of freedom

Multiple R-squared: 0.5542, Adjusted R-squared: 0.5493

F-statistic: 113.1 on 1 and 91 DF, p-value: < 2.2e-16

Here, the most important things are those listed underneath ‘Coefficients’, and especially beneath the word ‘Estimate’. These are the estimated values of the coefficients *a* (the intercept) and *b* (the slope) in the outline equation that we saw above.

We can also obtain a 95% confidence interval for each coefficient by using the following command.

confint(our\_model)

2.5 % 97.5 %

(Intercept) -5.1304047 3.8278811

Horsepower 0.1275048 0.1860669

Now, slotting in the estimated values of *a* and *b*, our regression equation for the relationship between Horsepower and Max.Price can be re-written in the form:

|  |
| --- |
| Max.Price = -0.65126 + (0.15679 × Horsepower) |

This means that, for any value of Horsepower, including values which we have not observed in our original data sample, we can make predictions about Max.Price.

And, by re-arranging the equation, we can also predict the value of Horsepower from any value of Max.Price.

R can handle predictions for us with the predict() command. The following will give us the Max.Price predictions for the existing values of Horsepower:

predict(our\_model)

1 2 3 4 5 6 7 8 9

21.298755 30.705905 26.315902 26.315902 31.960192 16.595180 26.002330 27.570189 26.002330

10 11 12 13 14 15 16 17 18

30.705905 45.600560 16.595180 16.595180 24.434472 16.595180 26.002330 25.218401 26.002330

19 20 21 22 23 24 25 26 27

46.384489 23.336971 21.455541 22.396256 13.773035 13.929821 15.027322 21.612327 15.027322

28 29 30 31 32 33 34 35 36

46.384489 13.773035 32.900907 9.226246 19.260539 14.400178 15.811251 17.379109 22.082684

37 38 39 40 41 42 43 44 45

21.298755 29.138047 7.971959 13.459463 24.434472 15.340894 21.298755 12.048391 18.790182

46 47 48 49 50 51 52 53 54

13.773035 19.417325 42.935201 28.354118 34.625551 24.434472 32.273764 12.205177 15.497679

55 56 57 58 59 60 61 62 63

25.061615 23.650543 39.329126 19.730897 33.371265 15.027322 21.298755 13.773035 31.019477

64 65 66 67 68 69 70 71 72

16.595180 22.866614 23.023399 24.434472 23.650543 16.595180 26.002330 26.002330 13.773035

73 74 75 76 77 78 79 80 81

10.950890 16.595180 24.434472 30.705905 26.002330 21.298755 12.675534 10.794104 13.459463

82 83 84 85 86 87 88 89 90

19.730897 10.323747 12.205177 20.514826 19.730897 20.985184 12.048391 16.438394 20.358040

91 92 93

27.256617 17.222324 25.688759

Remember that regression predicts the best-fit straight line through a cloud of observed points, so the predicted values of Max.Price for our existing Horsepower data will not be exactly the same as the actual observed values. Each prediction will have an element of error associated with it, which is called a residual.

The **residual** is simply the difference between the actual value in the data set and the predicted value from the model.

We can confirm this by first asking R to output the residuals of the regression model automatically.

our\_model$resid

1 2 3 4 5 6 7

-2.49875529 7.99409453 5.98409794 18.28409794 4.23980783 0.70481980 -4.30233038

8 9 10 11 12 13 14

-2.67018875 0.29766962 5.59409453 -2.90055993 1.70481980 -5.19518020 -7.63447202

15 16 17 18 19 20 21

1.80481980 -8.00233038 -6.61840120 -6.40233038 -4.88448911 -4.93697117 -4.35554113

22 23 24 25 26 27 28

7.10374385 -3.17303515 0.27017902 -0.32732184 2.78767303 1.37267816 -13.28448911

29 30 31 32 33 34 35

2.72696485 -11.70090718 -1.32624589 -7.36053942 -2.20017849 5.18874898 -2.17910938

36 37 38 39 40 41 42

3.21731553 3.50124471 -7.43804711 2.02804080 0.04053652 -1.73447202 0.45910649

43 44 45 46 47 48 49

-0.09875529 -2.84839095 -7.79018191 -2.77303515 -4.11732526 7.46479929 0.04588207

50 51 52 53 54 55 56

0.97444862 10.86552798 5.52623616 -3.10517678 -3.19767935 -6.36161537 -1.95054284

57 58 59 60 61 62 63

-6.82912647 15.16910307 46.62873531 -0.02732184 -6.39875529 -0.87303515 -1.11947715

64 65 66 67 68 69 70

-1.69518020 -4.56661366 -1.52339949 -2.43447202 -9.65054284 1.80481980 -6.50233038

71 72 73 74 75 76 77

-4.10233038 3.62696485 -1.05089009 -3.79518020 -3.03447202 -9.10590547 3.39766962

78 79 80 81 82 83 84

15.80124471 0.22446571 -1.29410426 -2.15946348 2.96910307 -0.32374675 -0.40517678

85 86 87 88 89 90 91

2.08517389 1.46910307 5.61481638 -2.54839095 6.26160563 2.04195973 -3.55661707

92 93

6.27767645 2.81124129

and then calculating the difference semi-manually:

Cars93$Max.Price - predict(our\_model)

1 2 3 4 5 6 7

-2.49875529 7.99409453 5.98409794 18.28409794 4.23980783 0.70481980 -4.30233038

8 9 10 11 12 13 14

-2.67018875 0.29766962 5.59409453 -2.90055993 1.70481980 -5.19518020 -7.63447202

15 16 17 18 19 20 21

1.80481980 -8.00233038 -6.61840120 -6.40233038 -4.88448911 -4.93697117 -4.35554113

22 23 24 25 26 27 28

7.10374385 -3.17303515 0.27017902 -0.32732184 2.78767303 1.37267816 -13.28448911

29 30 31 32 33 34 35

2.72696485 -11.70090718 -1.32624589 -7.36053942 -2.20017849 5.18874898 -2.17910938

36 37 38 39 40 41 42

3.21731553 3.50124471 -7.43804711 2.02804080 0.04053652 -1.73447202 0.45910649

43 44 45 46 47 48 49

-0.09875529 -2.84839095 -7.79018191 -2.77303515 -4.11732526 7.46479929 0.04588207

50 51 52 53 54 55 56

0.97444862 10.86552798 5.52623616 -3.10517678 -3.19767935 -6.36161537 -1.95054284

57 58 59 60 61 62 63

-6.82912647 15.16910307 46.62873531 -0.02732184 -6.39875529 -0.87303515 -1.11947715

64 65 66 67 68 69 70

-1.69518020 -4.56661366 -1.52339949 -2.43447202 -9.65054284 1.80481980 -6.50233038

71 72 73 74 75 76 77

-4.10233038 3.62696485 -1.05089009 -3.79518020 -3.03447202 -9.10590547 3.39766962

78 79 80 81 82 83 84

15.80124471 0.22446571 -1.29410426 -2.15946348 2.96910307 -0.32374675 -0.40517678

85 86 87 88 89 90 91

2.08517389 1.46910307 5.61481638 -2.54839095 6.26160563 2.04195973 -3.55661707

92 93

6.27767645 2.81124129

As you can see, the two sets of figures are identical.

To visualize the best­fit regression line on the scatterplot, we simply re­draw the scatterplot, using the same command as before, and then add the best-fit straight line with the abline() function.

plot(

Cars93$Horsepower,

Cars93$Max.Price,

pch = 18,

cex = 2,

col = "darkorchid4",

xlab = "Horsepower",

ylab = "Max. Price",

main = "Horsepower against Max. Price for 93 Cars in 1993"

)

abline(our\_model, lwd=2)

Remember: our\_model is the name that we gave our model. The statement lwd = 2 merely makes the best­fit line a bit thicker on the plot. You can omit this statement if you want to, or change the number to make it even thicker.

So, graphically, this is what our final regression model looks like. This is the same plot you saw earlier in the handout, except that now the shapes are dark-orchid colored and the model line superimposed.



*Assumptions of linear regression*

As well as assuming that there is a linear relationship between the two variables of interest, linear regression also assumes that the residuals of the regression model are approximately normally distributed.

Having fitted the regression model, we can check on this in the usual way, using normal QQ plots.

As you might remember from last week, the QQ-plot displays the quantiles of the observed data variable against the quantiles that we would theoretically expect if the data followed a normal distribution exactly. The plot contains two elements: a set of points (white bubbles) representing the observed data and a diagonal straight line representing the theoretical normal distribution.

If the data distribution is approximately normal, then we expect the data points to sit **more or less on top of the straight diagonal line**; any deviation from it should be minor and pretty random in its pattern. If we see larger and clearer deviations from the straight diagonal, then we should conclude that our data are not approximately normal.

qqnorm(our\_model$resid)

qqline(our\_model$resid)



On the whole, this QQ-plot doesn’t look too bad. There is some evidence of deviation from normality at the top­right of the diagonal. Note especially the final outlier point on this plot.

We could deal with this deviation in different ways. For example, we could apply a transformation to the data, such as taking the square root of the original values. Alternatively, we could remove any extreme outlying data points. Finally, we could use another form of regression modelling, but this is a more advanced topic (and quite a big one too...), and we won’t be covering it in this module.

Important: You should never exclude extreme (outlier) points from your data as a routine practice. Before excluding data, you should carefully consider why they are there and what effect they are having on the model. If you decide to exclude data points, this needs to be explained and justified in your report (paper, dissertation).

*Reporting the results*

We could write up the results as follows:

“A simple linear regression was calculated to predict maximum price based on horsepower. We found a significant relationship between horsepower and max. price, with a 0.16 unit increase in horsepower for every $1,000 increase in price, *R*2 = .16, *F*(1, 91) = 113.1, *p* < .001.”

It can also be helpful to include a graph with your results. For a simple linear regression, you can simply plot the observations on the x and y axis and then include the regression line and regression function.



Task 5: *Practice task*

Let’s practice! To give you more experience with correlations and linear regression, please replicate the analysis described above in the handout using the following two variables from the same sample data set (Cars93):

EngineSize (predictor) and Horsepower (outcome)

* + First, calculate the correlation between the two variables. Obtain and interpret the 95% confidence interval for the correlation coefficient. Note: Use a Spearman correlation, so that you get used to running the R code for intervals.
  + Perform the linear regression analysis.
  + Use the coefficients from the regression analysis to write a model equation expressing the relationship between EngineSize and Horsepower.
  + Check the model residuals for normality using a QQ-plot.

Note: The solution to this practice task is available on Moodle.

Task 6: *Testing categorical variables*

A few weeks ago, we reviewed two main types of variables, continuous variables and categorical variables.

With continuous variables, each case in our sample (each person, text, or whatever) is associated with a numerical score such as a percentage mark on a test, a proportional frequency (e.g., per thousand words), a measurement (e.g., height), a score on a seven­point scale, and so on. In such instances, we would typically expect that the number of distinct scores is quite large, sometimes equal to the number of cases in the sample.

Quite often, however, we may instead want to count how many individual cases fall into a set of categories. This might happen, for example, if we scored a test simply on the basis of pass/fail, or if we used a set of letter grades instead of numerical marks. It might also happen with questionnaires that have ordinal scales, if we choose not to re­code the scales as numbers.

This part of the handout aims to introduce you to the tests which we most commonly use for data involving such **categorical data**. We often refer to this sort of data simply as count data.

*Contingency tables*

Categorical data are normally represented in the form of a **contingency table**.

The smallest type of contingency table is a 2 x 2 table. That is, a table that represents two variables with two possible values each.

The following is an example of a 2 x 2 contingency table, representing smoking status of university students by level of study

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| Undergraduate | 20 | 80 |
| Postgraduate | 30 | 70 |

However, it is also possible to have much larger contingency tables, with more columns, or rows, or both. For example, here we have a 2 x 6 table.

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| UK |  |  |
| France |  |  |
| Germany |  |  |
| Italy |  |  |
| Spain |  |  |
| Greece |  |  |

*Two important points about contingency tables*

In contingency tables, the counts must be **true frequencies** (‘raw’ counts), not percentages or any other type of proportion.

It is usual to quote proportional figures as well when writing up, partly to avoid errors in interpretation when the sample sizes are unequal. However, they should never be used inside the statistical calculations themselves.

It does not normally matter which way round the tables are set up: most tests for count data are symmetrical. However, it is perhaps more common to display the main variable under investigation in the columns and the other variable (somewhat akin to a grouping variable) in the rows.

We make a basic distinction between ordered and unordered contingency tables.

**Ordered contingency tables** have an underlying rank order to at least one of the two variables.

Examples of this might be: questionnaire scales which range from ‘not very’ to ‘very’; coursework grades and degree classes; age groups; and any other groups that are based on measurements or frequencies.

**Unordered** **contingency tables** (also called nominal tables) do not involve these underlying rank orders.

Examples of unordered variables might be: nationality; TV region; religion; race; sexual orientation; political affiliation; etc.

It is also possible to have mixed variables. For example, an ordered questionnaire scale which also includes a ‘don’t know’ response; however, such mixed variables can be difficult to handle satisfactorily with the most easily accessible methods.

*Choosing an appropriate test for* unordered *tables*

For unordered tables we have three main options.

* Pearson’s chi­squared test
* The likelihood-ratio test
* Fisher’s exact test

The choice of test in these cases has been somewhat controversial.

**Pearson’s chi­squared test** is known to become increasingly liberal as the overall sample size gets bigger. That is, you will wrongly reject the null hypothesis rather more often than you should at the stated significance level. It has also been claimed that it requires a minimum expected frequency of five in all the cells of the contingency table in order to be reliable – the so­called ‘Cochran rule’ (Cochran, 1954). This is not universally accepted, and various other rules of thumb have also been proposed in the specialist literature. However, one broad consensus that does emerge is that **Fisher’s exact test** is a preferred alternative when the values in some cells of the table are small (Gries, 2005; Camilli, 1990).

There is also another test that can be used for count data: the **likelihood-ratio test** (also sometimes known as the log-likelihood test). This has been particularly popular in linguistics, for example, but does not originate there and it is not restricted to it. It became popular in linguistics following the publication of Dunning (1993), who argued that it was more accurate than Pearson’s chi­squared test on tables containing very large or very small frequencies. (And linguists deal quite often with relatively small word frequencies in samples of millions of running words of text, so this property can be important.) The likelihood-ratio test remains quite widely used.

It should be noted, however, that Fisher’s exact test is generally reckoned to be somewhat on the conservative side. In frequentist terms, this means that, in the long run, it leads us to wrongly retain, i.e. fail to reject, the null hypothesis rather more often than it should at the stated significance level.

The broad consensus, however, is that you should consider using Fisher’s exact test when you can, especially with 2 x 2 tables. Failing that, either the likelihood-ratio test or Pearson’s chi-squared test is the most convenient alternative choice.

*Choosing an appropriate test for* ordered *tables*

For 2 x *k* ordered tables, there are a number of options, including the use of rank correlations. It is also possible to analyse ordered tables in the same way as unordered tables. However, if analysed in this way, inferences about trends cannot be made. For instance, you could show that there is a statistically significant difference between passing and failing an exam in relation to different lengths of study period, but you could not show directly from this that, as people study for longer, they tend to pass more often than those who do not. The best that you could do is to identify those cells in the table that are contributing the most to the effect.

*Hypotheses*

These three tests are usually known as **tests of independence**.

They test the null hypothesis that the variables represented by the two edges of the table (e.g. study level versus smoking status, or country of residence versus smoking status) are independent of one another.

The alternative hypothesis is that they are associated. They are commonly also used for testing the equality of proportions (especially with a 2 x 2 table), but independence/association is strictly the underlying hypothesis.

*Entering count data into R*

Let’s use the following data for our first analyses.

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| UK | 9 | 141 |
| Poland | 46 | 104 |
| France | 19 | 131 |

R stores count data in a format known as a **data matrix**.

To enter the data from the table above, we can use the following command. This will create a matrix called our\_data.

our\_data <- matrix(c(9, 46, 19, 141, 104, 131), ncol = 2)

The statement ncol = 2 tells R that our table has two columns of data; R will work out the number of rows for itself.

The actual counts are entered inside the pair of brackets that follow the letter ‘c’, with a comma separating each value. The key thing that you need to remember here is that you must enter your numbers **column by column**, and NOT row by row.[[3]](#footnote-4)

So, for the table shown above, the correct number sequence is: 9, 46, 19, 141, 104, 131.

If you tried to enter the numbers row by row, i.e. 9, 141, 46, 104, 19, 131, then R would construct a table that corresponds to the following, which is clearly WRONG.

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| UK | 9 | 104 |
| Poland | 141 | 19 |
| France | 46 | 131 |

If you can remember this basic rule, then everything else is quite straightforward. But it is very easy to forget, so I recommend that you always inspect the data table that you have entered inside R, before you carry on to do any tests on it.

So, let’s inspect the matrix to make sure we entered the data correctly.

our\_data

[,1] [,2]

[1,] 9 141

[2,] 46 104

[3,] 19 131

*Adding row and column names*

As you can see, the matrix is missing names for rows and columns. Adding names is optional, of course, as long as you remember which is which.

The following command will add names for rows and columns.

rownames(our\_data) <- c("UK", "Poland", "France")

colnames(our\_data) <- c("Smoker", "Non-smoker")

This is what our data matrix now looks like.

our\_data

Smoker Non-smoker

UK 9 141

Poland 46 104

France 19 131

*Running tests on the contingency table*

To run chi­squared or Fisher tests on the contingency table, we use the following commands.

First, the chi-squared test. The option correct = FALSE in the chisq.test() command turns off what is known as the ‘Yates correction’. This only applies to 2 x 2 tables, so it has no effect with larger ones. The advantage of using the Yates correction on a 2 x 2 table is a bit controversial, so I prefer to switch it off as a default.

x.test <- chisq.test(our\_data, correct=FALSE)

x.test

Pearson’s Chi-squared test

data: our\_data

X-squared = 35.548, df = 2, p-value = 1.909e-08

And now the Fisher test.

f.test <- fisher.test(our\_data)

f.test

Fisher’s Exact Test for Count Data

data: our\_data

p-value = 2.773e-08

alternative hypothesis: two.sided

Note: You can actually type just the right­hand side of the above commands, i.e. the part after the assignment operator <- . If you wrote chisq.test(our\_data, correct=FALSE) or fisher.test(our\_data) you would get the results immediately. However, it is better to do it in the above way, as we shall want to access some other ‘hidden’ elements of the test results later on.

Note that, for 2 x 2 tables, the Fisher test command will also output the alternative hypothesis in terms of the **Odds Ratio** (OR), and it will also output the sample estimate and 95% confidence interval for the OR.

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| Undergraduate | 20 | 80 |
| Postgraduate | 30 | 70 |

For example, if we ran the Fisher test for the 2 x 2 table immediately above, your input would include the 95% confidence levels.

two\_by\_two <- matrix(c(20, 30, 80, 70), ncol = 2)

two\_by\_two

fisher.test(two\_by\_two)

Fisher’s Exact Test for Count Data

data: two\_by\_two

p-value = 0.1412

alternative hypothesis: true odds ratio is not equal to 1

95 percent confidence interval:

0.287110 1.171871

sample estimates:

odds ratio

0.5849237

The Odds Ratio is a special kind of effect size. It is very commonly used in medicine and epidemiology, but it has sometimes been adopted in other fields, too.

As its name suggests, it is a ratio, i.e. one number divided by another, that compares the odds in favour of something occurring under one condition with the odds in favour of it occurring under another condition.[[4]](#footnote-5)

In the medical context, the typical comparison is the odds of being infected when exposed to a disease versus the odds of being infected when not so exposed. In our example table, it is the odds of being a smoker given that you are an undergraduate, compared with the odds of being a smoker given that you are a postgraduate.

When the odds are the same in both conditions, the OR is exactly 1.

If the odds are greater under the first condition, the OR will be larger than 1.

And, if the odds are greater under the second condition, the OR will be smaller than 1.

So, if you are using the 95% CI to draw an inference, remember that the null hypothesis value for an OR will typically be 1, not the zero that we tend to use in other contexts (such as a difference in means).

Note also that the R implementation of the Fisher test does not like tables with large dimensions and very large numbers (because the Fisher calculation is very computationally intensive). It will often fail in such cases. In these instances, it is best to use the chi­squared or likelihood-ratio test instead.

*Likelihood-ratio test*

To run the likelihood-ratio test (i.e. the log­likelihood test), we must instead type the following three lines of code.

library(MASS)

lr\_test <- loglm( ~ 1 + 2, our\_data)

And our output looks like this.

lr\_test

Call:

loglm(formula = ~1 + 2, data = our\_data)

Statistics:

X^2 df P(> X^2)

Likelihood Ratio 35.25742 2 2.207746e-08

Pearson 35.54845 2 1.908766e-08

The first line of the ‘Statistics’ gives you the results for the likelihood-ratio test, i.e. the test statistic, degrees of freedom, and p­value.

The second line gives you the results for the ordinary (Pearson) chi­squared test, which should be identical to those obtained by running chisq.test in the way shown earlier.

*Effect sizes*

As with most of the common statistical tests, the key item of information that results from any of these tests for count data is a p­value. However, as we have said before, a p-value does not tell you how big or important a difference or a relationship is: it merely tells you the probability (in the long run) of encountering an equal or larger statistic, if the null hypothesis happens to be true each time.

To show how big the difference or relationship is, you need to quote an effect size as well. With differences between two means, which we looked at last time, it often suffices simply to quote the actual difference. However, with contingency tables (especially those larger than 2 x 2), a simple ‘surface’ statistic like this is not really available. Instead, we have to use a special effect size measure. We have already seen the Odds Ratio, which we can use with a Fisher test for a 2 x 2 table. For chi­squared tests, there are two other common effect-size calculations, depending on the dimensions of your contingency table.

For a 2 x 2 table, the effect size calculation is known as **phi** (or ) and is performed using the following formula:

where is the chi­squared test statistic (not the p­value) and N is the sum of all the counts in the contingency table.

In R, assuming that your data matrix is called our\_data and the output of the chi­squared test is stored under the name x.test (as in the examples above), this would be entered as:

sqrt(x.test$stat / sum(our\_data))

X-squared

0.2810632

For a matrix larger than 2 x 2, the effect size calculation is known as **Cramer’s V** and is performed using the following formula, of which the previous formula for phi is merely a simplification.

Again, is the chi­squared test statistic and N is the sum of counts in the contingency table.

In the denominator, min(r,c) − 1 simply means that you count how many columns and rows your table has and insert the smallest of these two numbers, minus one, into the equation. For example, if your table has three rows and five columns, then the value of min(r,c) − 1 is 3 − 1 = 2.

In R, with the same naming assumptions as above, the code for this would be:

sqrt(x.test$stat / (sum(our\_data) \* (min(dim(our\_data)) - 1)))

X-squared

0.2810632

Note that these lines of code produce a number with ‘X-Squared’ printed on top of it. Unfortunatley, it is not possible to remove that label from the output, but please note that this is NOT a chi-squared value; it is **Cramer’s V**.

Both these effect sizes (phi and Cramer’s V) can be broadly interpreted as follows:

* 0 to 0.1 is a negligible effect;
* 0.1 to 0.3 is a small effect;
* 0.3 to 0.5 is medium-sized effect; and
* values greater than 0.5 are large effects.

But bear in mind that these are merely conventional rules of thumb and may need to be calibrated differently in certain contexts.

*Analysing the Residuals*

The chi­squared test is fundamentally about the difference between observed and expected counts. Indeed, the formula for the chi­squared test statistic is quite simply:

where *O* = the observed (i.e. actual) count in each cell of the contingency table and *E* = the expected count in each cell under the null hypothesis. (The ∑ symbol simply means ‘add them all up’ for all the cells in the table.) *E*, for each cell, can be calculated straightforwardly by the following formula:

where *ni* is the sum of counts in the relevant row, *nj* is the sum of counts in the relevant column, and *N* is the sum of the counts in the table as a whole. All this is doing is partitioning evenly the overall count for the table, considering the fixed sub-totals for the columns and rows respectively. In other words, it is what we would expect to see if there were no association whatsoever between the two variables represented by the table.

The value for each cell of the table is called the Pearson residual, and it can help us to identify which individual cells are contributing most to the overall result. So it is simply the difference between the observed and expected values for a cell, scaled by the square root of the expected value. The residuals are particularly valuable in tables larger than 2 x 2, where it can be much harder to appreciate what is going on with the naked eye; but, even with 2 x 2 tables, they can still be helpful, especially if the sample sizes in the columns/rows are quite different.

Basically, the larger the Pearson residual (be it positive or negative), the greater the contribution of that particular cell to the overall effect. We are usually looking for residuals of at least 2 or larger. These are the cells that deviate most from the values to be expected under the null hypothesis.

To look at the Pearson residuals in R, again assuming that you saved the test output as x.test, you can type:

x.test$residuals

Smoker Non-smoker

UK -3.154433 1.3994046

Poland 4.295399 -1.9055723

France -1.140965 0.5061676

The cells in this table correspond to the cells in the original data table that is being tested. In this example, we can see two residuals larger than our rule-of-thumb minimum of 2 (see highlights above). These are in the first two rows of the first column. In the original data table, these cells are those containing the counts of smokers in the UK and Poland. The residuals, therefore, tell us that the number of smokers we observed in the UK is smaller than we would expect under the null hypothesis (it has a negative residual) and the numbers of smokers in Poland is larger than we would expect under the null hypothesis (it has a positive residual).

We need to be careful with our phrasing when writing about residuals. These are not the frequencies themselves, or proportional frequencies, or even the raw differences between the observed and expected frequencies (because they are scaled by the square root of the expected value). They are merely an indication of those frequencies which are substantially smaller or larger than we would have expected to see if the null hypothesis of ‘independence’ or ‘no association’ were true. And we need to avoid saying ‘significantly’ in this context (rather than e.g. ‘substiantially’) because the residuals are not themselves a test of statistical significance.

You can also, incidentally, look at the expected values for a table by using:

x.test$expected

Smoker Non-smoker

UK 24.66667 125.3333

Poland 24.66667 125.3333

France 24.66667 125.3333

You might need to use this latter command if you wanted to check for the ‘Cochran rule’ mentioned earlier in the notes.

*Reporting results*

If you are using APA style, the chi-squared could be reported as follows.

“A chi-square test of independence was performed to examine the relation between smoking status and country of residence. The relation between these variables was significant, χ2(2) = 35.55, p < .001, but the effect size was small, Cramer’s V = .14. The analysis of the residuals suggests that the number of smokers in the UK is smaller than we would expect under the null hypothesis, and the numbers of smokers in Poland is larger than we would expect under the null hypothesis. A contingency table for the data is below.”

|  |  |  |
| --- | --- | --- |
|  | Smoker | Non­smoker |
| UK | 9 | 104 |
| Poland | 141 | 19 |
| France | 46 | 131 |

If using the likelihood-ratio test, you should probably write ‘likelihood-ratio χ2’ for clarity. The relevant effect size (phi or Cramer’s V) can be added after the p-value.

Fisher’s exact test has no test statistic as such, and it thus requires only the p-value to be reported. But, for 2 x 2 tables, you might also wish to report the odds ratio (OR) and its 95% CI, as in the following example.

*p* = .14, OR = .58, and 95% CI [.29, 1.17]

You still need to make it clear somewhere that you used Fisher’s exact test; don’t just write down a p-value without saying where it came from.

1. Note that we can also use correlations for binary and ordinal data, though we do not address these special cases on this module. [↑](#footnote-ref-2)
2. A transformation means doing something like taking the square root of the data values. This sometimes helps to make the data approximate more closely to a normal distribution, but it can also make the results of an analysis harder to interpret, so is not always to be recommended. [↑](#footnote-ref-3)
3. Actually, you can enter them row by row, but you must then remember to add an extra flag (byrow = T) to the command. [↑](#footnote-ref-4)
4. You will probably know the word ‘odds’ already from the context of sports betting – especially horse racing. The odds in favour of something are simply the probability of its occurring, divided by the probability of its not occurring. (Bookmakers tend instead to talk in terms of the odds *against*, which flips that equation on its head...) [↑](#footnote-ref-5)