Class 2: Regression and classification

Andrew Parnell, School of Mathematics and Statistics,
University College Dublin

Learning outcomes

- Be able to understand the structure of regression and classification models
- Know how to read and interpret the output of a statistical model
- Be familiar with some of the extensions to basic regression and classification models

Why regression and classification?

- t-tests are only really useful when you have a continuous outcome variable and one discrete variable with two groups (e.g. treatment vs control)
- For almost any real life situation you have multiple variables of all different types
- For these situations you need a *statistical model*
- A statistical model allows us to perform *probabilistic prediction* of the outcome variable from the remaining variables, and/or to explain how the other variables are causing the outcome variable to change

Regression vs Classification: what's the difference?

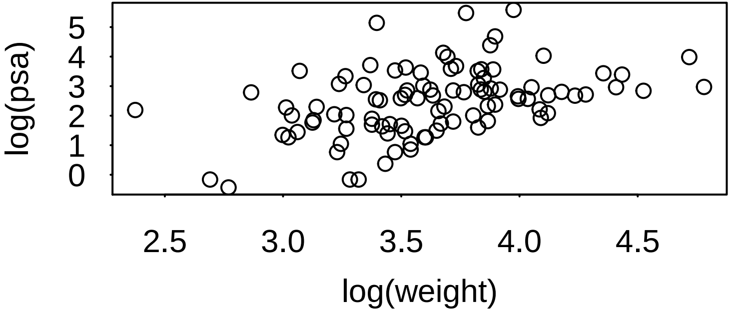
- In regression we have a single *continuous* outcome variable and lots of other variables which we think might be good predictors of the outcome
- In classification we have a single *discrete* outcome variable and lots of other variables
- In the machine learning literature this is often known as *supervised learning*
- Situations where there are multiple outcome variables are beyond the scope of this course

Response and explanatory variables

- The outcome variable is more commonly known as the *response* variable
- The other variables which we think might be good predictors of the response variable are called the *explanatory variables* (though be careful with causation)
- We will use these words from now on, but beware there are lots of other terms in the literature

A basic regression model

- Let's go back to the prostate cancer data
- Recall the key outcome variable is `lpsa` the log of the prostate specific antigen value. This is our response variable
- Suppose we had one explanatory variable `lweight`



Creating the model

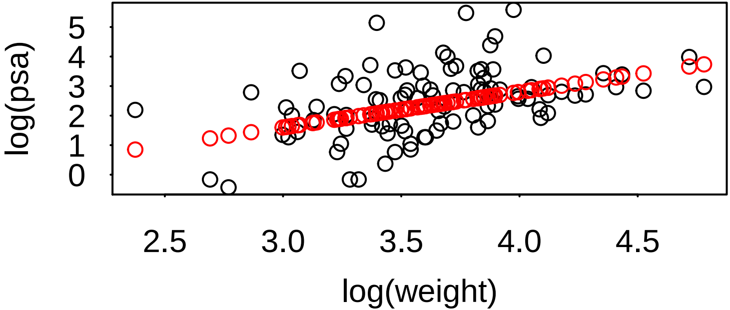
- Looking at the plot, there may be a positive, linear relationship between log(weight) and log(psa)
- Perhaps we can create a prediction model that allows us to predict log(psa) from log(weight)
- Suppose, for each patient we multiplied the log(weight) value by 1.2 and then subtracted the value 2 so:

$$prediction = 1.2 \times \log(weight) - 2$$

- If we do this repeatedly for every value in the data set we get ...

A first model

```
prediction = 1.2 * prostate$lweight - 2
plot(prostate$lweight, prostate$lpsa,
     xlab = 'log(weight)', ylab= 'log(psa)')
points(prostate$lweight, prediction, col='red')
```



Refining the model

- Is this model any good?
- How might we measure how close our predictions are to the truth?
- How can we choose the values (here 1.2 and -2) better?

Getting R to do the work

- Luckily the R function `lm` will do the work for us

```
model_1 = lm(formula = lpsa ~ lweight, data = prostate)
summary(model_1)
```

```
...
coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.7586     0.9103  -1.932   0.0564 .
lweight       1.1676     0.2491   4.686 9.28e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 1.046 on 95 degrees of freedom
Multiple R-squared:  0.1878,  Adjusted R-squared:  0.1792
F-statistic: 21.96 on 1 and 95 DF,  p-value: 9.276e-06
...
```

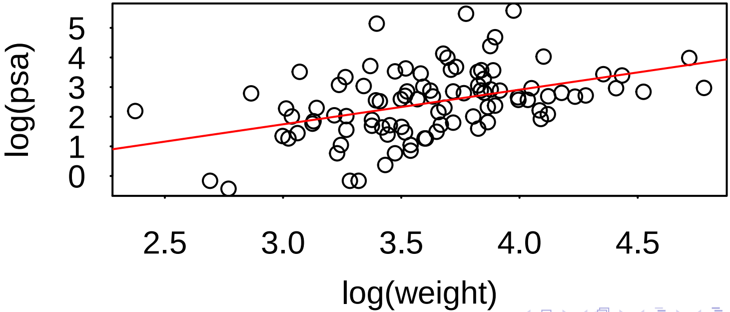
Background details

- The two values here are the *y*-intercept and the slope of the line. They are commonly known as the *regression coefficients*
- R chooses these coefficients by minimising the vertical distances between the black and the red points
- A key assumption in the model is that these vertical distances (known as *residuals*) are normally distributed
- R uses this assumption to run t-tests on the parameters, which you can see the results of in the `summary` output

Plotting the fit

One way is to type `plot(model_1)` which gives residual diagnostics. A quick plot of the fitted line via:

```
plot(prostate$lweight, prostate$lpsa,
     xlab = 'log(weight)', ylab= 'log(psa)')
abline(model_1, col='red')
```



Expanding the model with two explanatory variables

Suppose we wanted to use two explanatory variables, `lweight` and `age`:

```
model_2 = lm(formula = lpsa ~ lweight + age,
             data = prostate)
summary(model_2)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.897709   1.119033  -1.696   0.0932 .
lweight      1.147487   0.267094   4.296 4.23e-05 ***
age          0.003318   0.015369   0.216   0.8295
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘

Residual standard error: 1.051 on 94 degrees of freedom
Multiple R-squared:  0.1882,	Adjusted R-squared:  0.1709
F-statistic: 10.89 on 2 and 94 DF,  p-value: 5.558e-05
```

Expanding the fit even more

```r
model_3 = lm(formula = lpsa ~ . - train, data = prostate)
summary(model_3)
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.181561   1.320568   0.137  0.89096
lcavol        0.564341   0.087833   6.425 6.55e-09 ***
lweight       0.622020   0.200897   3.096  0.00263 **
age          -0.021248   0.011084  -1.917  0.05848 .
lbph          0.096713   0.057913   1.670  0.09848 .
svi           0.761673   0.241176   3.158  0.00218 **
lcp          -0.106051   0.089868  -1.180  0.24115
gleason       0.049228   0.155341   0.317  0.75207
pgg45         0.004458   0.004365   1.021  0.31000
```

Multiple regression

- When you have lots of explanatory variables this is known as *multiple regression*
- You can still use the values in the `Estimate` column to create predictions of `lpsa` by multiplying and adding up
- Beware the p-values as before: they might be highly significant but still a very poor model
- R gives you two other useful statistics:
  - The R-squared which measures the proportion of variation in the response variable explained by the explanatory variables
  - The residual standard error which measures how far away the data points are from the fitted line

Dealing with interactions

- Our explanatory variables will often interact with each other to affect the response variable
- The usual way to deal with interactions is to create *new explanatory variables* by multiplying them together. `lm` does this for you:

```
model_4 = lm(formula = lpsa ~ lweight + age + lweight:age,
summary(model_4)
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -9.97325    8.45553  -1.179    0.241
lweight      3.45163    2.40620   1.434    0.155
age          0.12575    0.12800   0.982    0.328
lweight:age -0.03481    0.03613  -0.964    0.338

Residual standard error: 1.051 on 93 degrees of freedom
Multiple R-squared:  0.1962,  Adjusted R-squared:  0.1703
F-statistic: 7.566 on 3 and 93 DF,  p-value: 0.0001391
```

Final remarks on regression models

- There is lots of research on regression models of all different types
- The vast majority of them involve creating a set of coefficients to multiply the explanatory variables by and then adding everything up
- It becomes very hard to plot the predictions in large and complex models
- It's very important to check the model diagnostics using `plot` and to look at the R-squared and residual standard error values

Classification models

Intro to classification models

- Returning to the South African heart rate data, recall that here we are interested in predicting whether someone has CHD or not
- We have explanatory variables including adiposity, alcohol use, age, etc
- CHD is a discrete binary variable (1 or 0). It thus makes more sense to try and predict a probability of CHD i.e. a value between 0 and 1, rather than CHD itself
- If we use our previous approach to guess coefficients for the different explanatory variables we will run into problems with values going outside 1 or 0

The logit transformation

- Suppose we wanted to predict CHD from age
- We might come up with the model:

$$Prob(CHD) = 0.06 \times age - 2$$

- Thus if someone has an age of 40 they have probability 0.4
- But if someone has an age of 20 they have probability -0.8. Oh dear!
- Instead use a transformation, such as the *logit*

$$Prob(CHD) = \frac{\exp(0.06 \times age - 2)}{1 + \exp(0.06 \times age - 2)}$$

- This transformation *garuantees* the values will be between 0 and 1 - try it!

About classification models

- Rather than try to predict a continuous response variable, classification models aim to find the probability that an observation is in a particular class
- Underneath the hood though, they are exactly like regression models with coefficients applied to each of the explanatory variables before adding everything up
- We then use a clever transformation (such as the logit, but there are others) to turn it into a probability
- Rather than the normal distribution, we use the *binomial* distribution to judge how close the observations are to the predictions and hence estimate the missing coefficients
- The R function `glm` will fit a classification model for us

Example: SA Heart rate

```
model_1 = glm(chd ~ age, data = SA, family = 'binomial')
summary(model_1)
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.521710   0.416031  -8.465  < 2e-16 ***
age          0.064108   0.008532   7.513 5.76e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 596.11  on 461  degrees of freedom
Residual deviance: 525.56  on 460  degrees of freedom
AIC: 529.56
```

Understanding the output

- The output here is much less helpful
- We have the coefficient values, but this is before the logit transformation so not particularly useful
- We have the p-values of the coefficients but we should be wary of these
- The other values (deviance etc) aren't particularly helpful
- AIC we'll talk about in the next class
- In fact, to judge the performance of the model we need to do a lot more work!

Extending the model

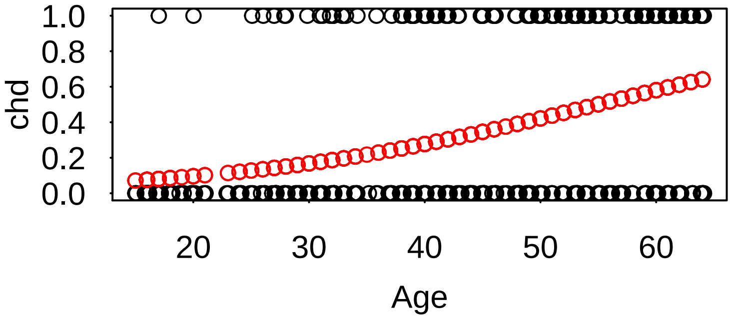
We can extend to multiple explanatory variables in exactly the same way as before:

```
model_2 = glm(chd ~ age + adiposity + age:adiposity, data =
summary(model_2)
```

```
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)   -4.6909894  1.3851412  -3.387 0.000708 ***
age            0.0811012  0.0300150   2.702 0.006892 **
adiposity      0.0583492  0.0596028   0.979 0.327596
age:adiposity -0.0009184  0.0012051  -0.762 0.446000
```

Plotting the fitted model

```
plot(jitter(SA$age), SA$chd, ylab = 'chd', xlab = 'Age')
points(SA$age, model_1$fitted.values, col='red')
```



Regularisation and shrinkage

- When you have lots and lots of explanatory variables, the model can become very slow or might not fit at all
- Worse, we might have lots of spurious small p-values without any predictive power
- It makes sense to remove or reduce some of the coefficients on the explanatory variables if we think their effect is over-stated
- One way of doing this is via *regularisation*, where we set some of the values to zero, another is via *shrinkage* where we reduce the values (shrink them towards zero)

Lasso and Ridge

- The R package `glmnet` will perform shrinkage and regularisation for both regression and classification
- The *Lasso* model imposes a restricted sum on the absolute value of all of the coefficient values
- The *Ridge* model imposes an assumption that all of the coefficient values come from a normal distribution with some small standard deviation
- We will play with some of these models later

More advanced classification approaches

- Much like regression, classification models have a long literature
- However, classification models tend to be more complicated as there are transformations involved (e.g. logit) and often multiple response variables (i.e. more than two categories for the response)
- Sometimes you have the choice between using a discrete response variable or a continuous one. I would always pick the continuous one: in general *regression models perform better than classification models*

Summary

- We now know how to implement regression and classification models in R
- We know how to interpret the output of some regression models
- We're familiar with some of the more advanced concepts in regression and classification