The General Linear Model - Regression Part 2

Unit Coordinator: Dr Danna R. Gifford
(danna.gifford@manchester.ac.uk)

Original author: Prof Andrew J. Stewart
(drandrewjstewart@gmail.com)

Previously

We went from variance to covariance to correlation to simple regression.

We discussed how regression can be thought of as prediction.

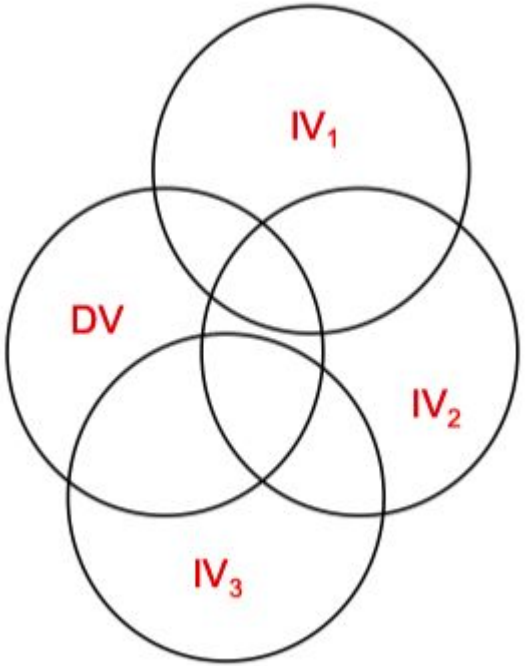
With OLS regression, we saw how to build different models of our data and test how good each is via measuring the extent to which they minimise the residuals - i.e., the difference between the observed data and our linear model.

Multiple Regression

Multiple regression is just an extension of simple linear regression - but rather than having one predictor we have several.

Our goal is the same though - to end up with an equation for a straight line that is the best fit to our data - in other words, we want to minimise the residuals.

Defining the Problem



Here we have a Dependent Variable (our Outcome) and three Independent Variables (our Predictors). To what extent do each of our Independent Variables predict our Outcome? Do we need to worry about our Independent Variables being (too) correlated with each other?

Data Considerations

Sample size - do we have enough power to detect our minimal effect size of interest?)

Do we need to worry about issue related to multicollinearity and singularity?

Ensure our residuals are approximately normal - and that we don’t have any issues around skewness, nonlinearity, or homoscedasticity.

Do we have any outliers and/or influential cases?

How big an N do we need?

For testing the regression model, $N \geq 50 + 8k$ (where k = no. of predictors)

For testing individual predictors, $N \geq 104 + k$

Usually we want to test both so calculate both equations and choose the one that produces the largest number.

For example, with k=6 we require N=98 to test the regression and N=110 to test the individual predictors. We select 110.

Multicollinearity and Singularity

Multicollinearity: when two or more variables are highly correlated (tested by examining VIF value for each variable).

Singularity: redundancy (e.g., one of the variables is a combination of two or more of the other IVs).

We can use collinearity diagnostics to see if we have a possible problem.

Assumptions: no multicollinearity among predictors

VIF stands for Variance Inflation Factor. Essentially, it tells us about when we have to worry about (multi)collinearity.

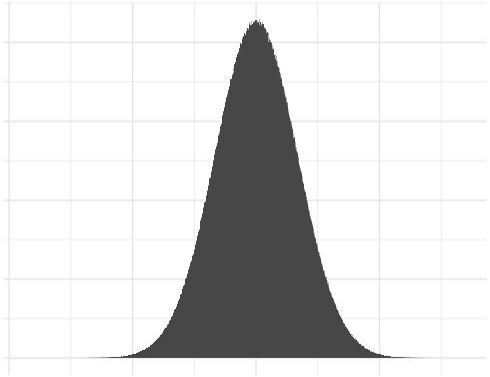
We can ask for VIF to any model in R by using the function `vif()` in the `car` package.

So when do you worry?

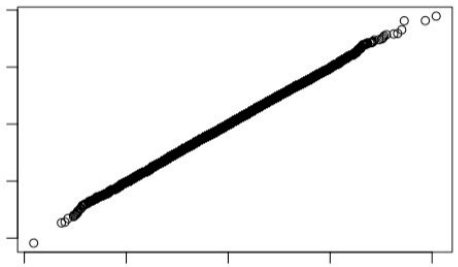
As a rule of thumb VIF greater than 10 suggests a multicollinearity issue (although greater than 5 has been suggested too - more conservative).

Assumptions: normality of residuals

For every predicted score, the observed scores around that prediction should be normally distributed (i.e., normally distributed error).



Can be investigated using a Q-Q plot. Diagonal line indicates normality of residuals.



Assumptions: linearity

The relationship to be modelled must be linear.

That is, an increase in the scores of a predictor should be followed by a increase in the outcome and vice versa.

There must be a uniform relationship between the fitted values and the residual error that can be captured by a straight line.

Minor violation of this assumption is not dire – it weakens the power of the analysis to capture the relationship between the variables.

Assumptions: homoscedasticity

Standard deviations of errors of prediction should be approximately equal for all predicted scores.

That is, the amount of error is the same at different predicted values of the outcome.

Violation is not terminal, but does weaken the analysis.

Can be corrected by using weighted (generalised) least squares regression instead of OLS regression.

Assumptions: outliers

Outliers are cases that do not fit well with the regression model (cases with large residuals).

Like correlation, regression models are sensitive to outliers so it makes sense to consider removing/replacing them to improve the model.

Bear in mind that even if you remove them they may be theoretically interesting.

Influential cases can be detected easily via Cook's distance (part of our diagnostic plots) using the performance package.

The Linear Model with Multiple Predictors

We are interested in whether house prices in 250 regions of the UK can be predicted by:

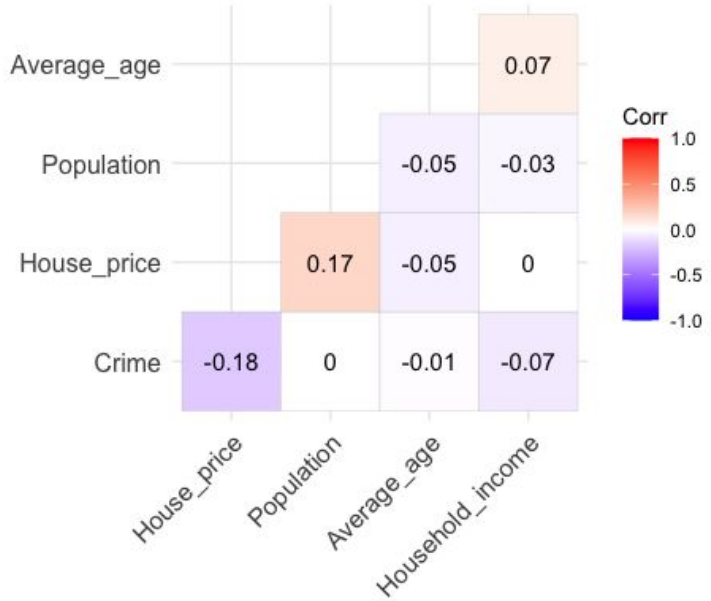
(a) Population size
(b) Crime rate (per 10,000 people)
(c) Average age of people in the region
(d) Average household income in the region.

Including our predictor and a column identifying our Regions, our datasets consists of 6 variables.

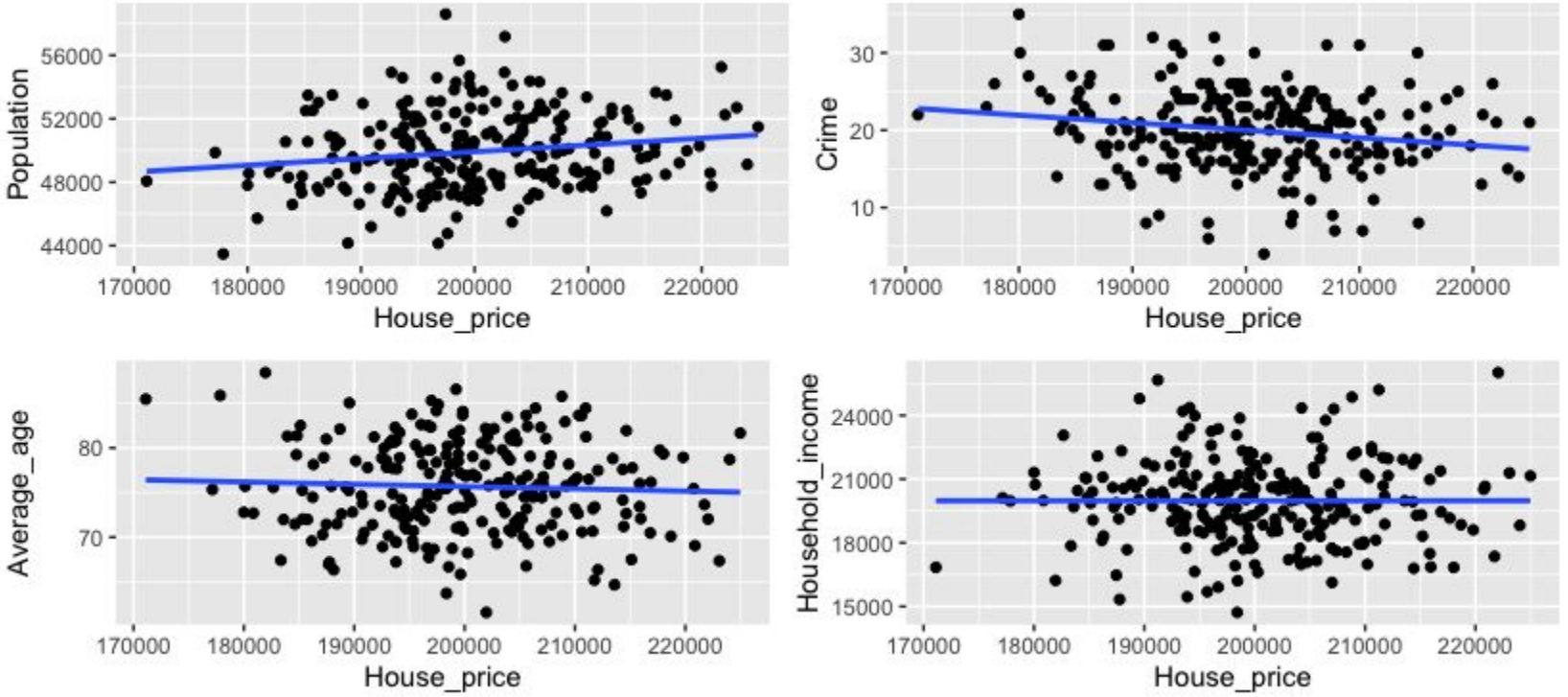
```
> str(my_data)
tibble [250 × 6] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Region          : Factor w/ 250 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ House_price     : num [1:250] 193735 201836 191643 215952 203295 ...
 $ Population      : num [1:250] 49004 48307 50379 53664 45481 ...
 $ Crime           : num [1:250] 14 25 19 17 22 32 21 21 20 25 ...
 $ Average_age     : num [1:250] 72.7 78.1 71.4 72.1 76.1 ...
 $ Household_income: num [1:250] 20843 19130 20411 16863 19964 ...
```

Let's build a correlation plot - using the function `ggcorrplot()` from the the package `ggcorrplot`.

```
corr <- cor(dplyr::select(my_data, -Region))

ggcorrplot(corr, hc.order = TRUE,
           type = "lower", lab = TRUE)
```



Each Predictors vs. Our Outcome



First we will build a linear model with all 4 predictors, then a second model with just the intercept (i.e., the mean) - we then compare them - is the model with the 4 predictors a better fit to our data than the model with just the mean?

```
> model0 <- lm(House_price ~ 1, data = my_data)
> model1 <- lm(House_price ~ Population + Crime + Average_age + Household_income,
data = my_data)
> anova(model0, model1)
Analysis of Variance Table

Model 1: House_price ~ 1
Model 2: House_price ~ Population + Crime + Average_age + Household_income
  Res.Df        RSS Df  Sum of Sq      F  Pr(>F)
1    249 2.3069e+10
2    245 2.1622e+10  4 1446836735 4.0985 0.00311 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The F-ratio comparing our two models is 4.0985 indicating our model with all the predictors is a better fit than our model with just the intercept (the mean). We then need to get our parameter estimates using the function `summary()`

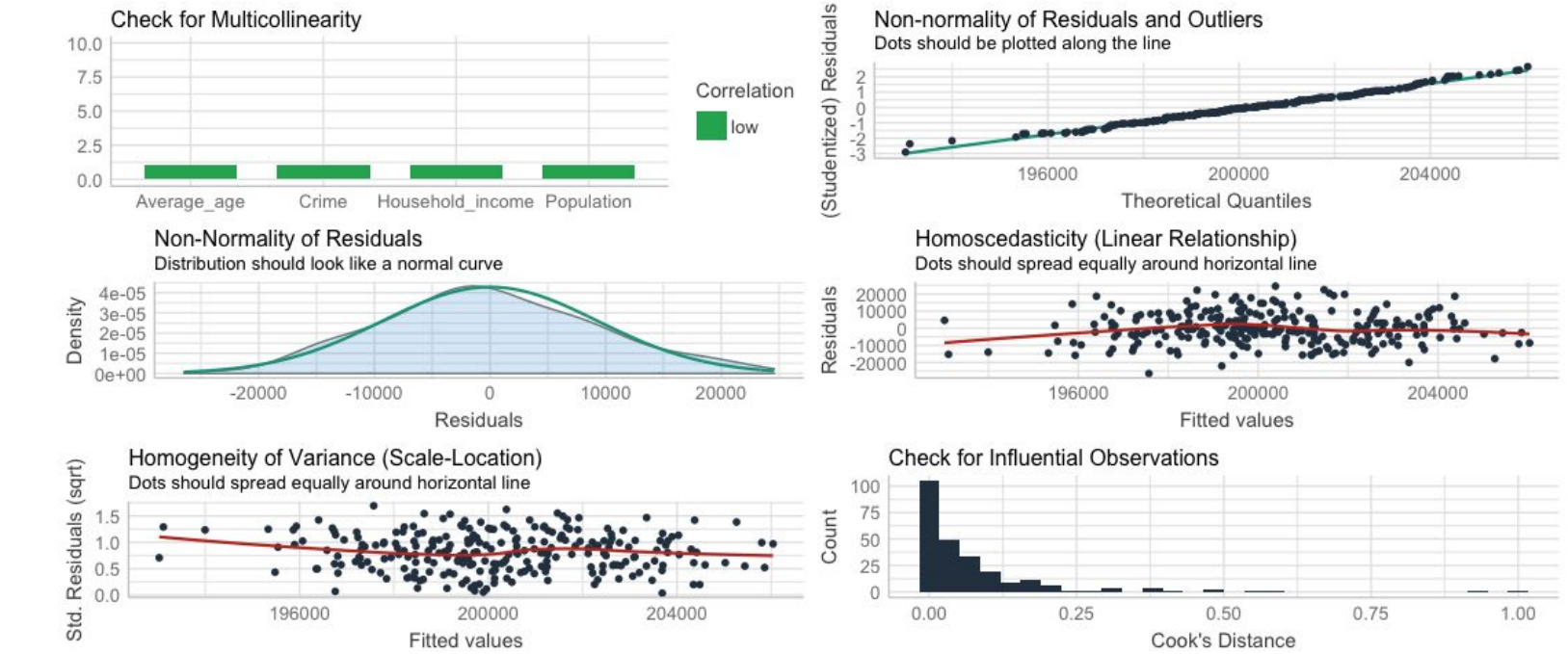
```
> summary(model1)

Call:
lm(formula = House_price ~ Population + Crime + Average_age +
    Household_income, data = my_data)

Residuals:
     Min       1Q   Median       3Q      Max
-26460.7  -6011.9   -386.4   6331.8  24591.6

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       1.807e+05  1.680e+04  10.754  < 2e-16 ***
Population        6.577e-01  2.453e-01   2.682  0.00782 **
Crime            -3.358e+02  1.153e+02  -2.913  0.00391 **
Average_age      -8.218e+01  1.186e+02  -0.693  0.48915
Household_income -1.957e-02  3.033e-01  -0.065  0.94861
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9394 on 245 degrees of freedom
Multiple R-squared:  0.06272, Adjusted R-squared:  0.04741
F-statistic: 4.098 on 4 and 245 DF,  p-value: 0.00311
```

Here we have our parameter estimates and the t-tests associated with our predictors.

Here are the R-squared and Adjusted R-squared values (which reflects the number of predictors in our model).

Checking the assumptions

```
> check_model(model1)
```



```
# Notice that Average_age and Household_income do not seem to predict house prices
# Let's drop them in model2
model2 <- lm(House_price ~ Population + Crime, data = my_data)
anova(model2, model1)
```

```
> anova(model2, model1)
Analysis of Variance Table

Model 1: House_price ~ Population + Crime
Model 2: House_price ~ Population + Crime + Average_age + Household_income
  Res.Df        RSS Df Sum of Sq      F Pr(>F)
1    247 2.1666e+10
2    245 2.1622e+10  2  43401593 0.2459 0.7822
```

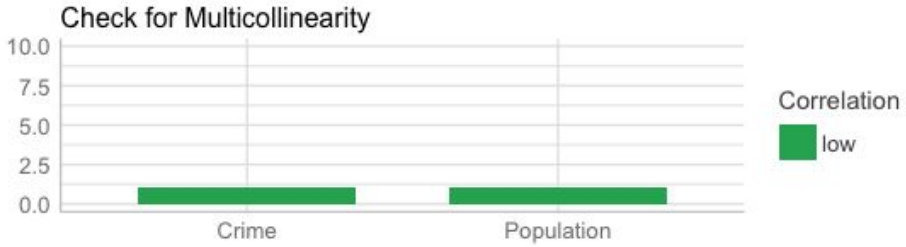
OK, so the models do not differ significantly by this test - we can use another measure of goodness-of-fit - AIC (Aikaike Information Criterion). AIC tells us how much information in our data is not captured by each model - lower values are better - can only be interpreted in a relative sense (i.e., comparing one model to another).

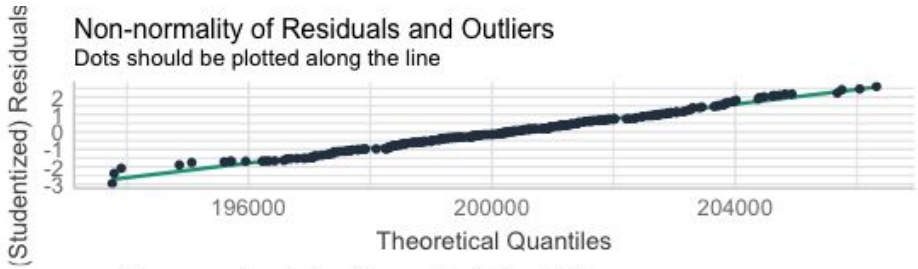
```
> AIC(model1)
[1] 5290.354
> AIC(model2)
[1] 5286.855
```

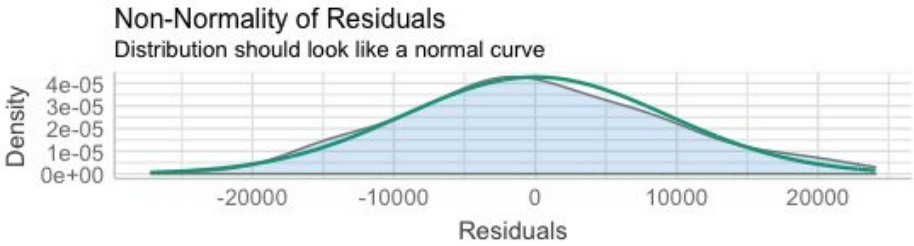
We defined `model2` as having just two predictors - as `model2` has the lower AIC value (so more information in our data is explained by `model2` than by `model1`), we would be justified in selecting that as our ‘best’ model. AIC penalises models with increasing number of parameters (but not as much as BIC) so gives us a good trade-off of fitting our data and model complexity.

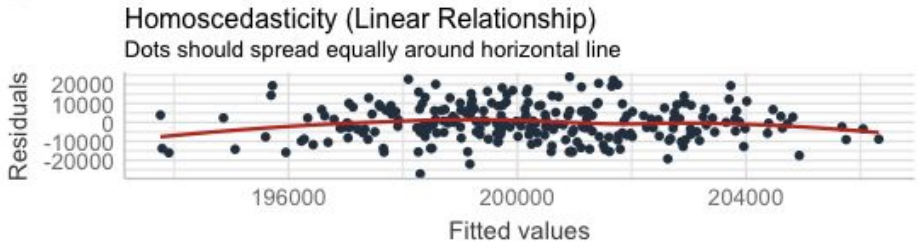
Let’s look at our diagnostic plots for `model2`.

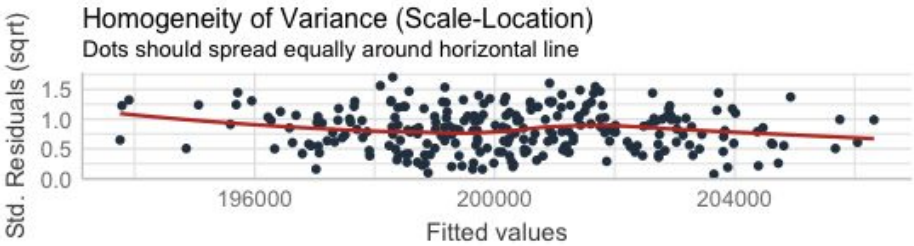
```
> check_model(model2)
```

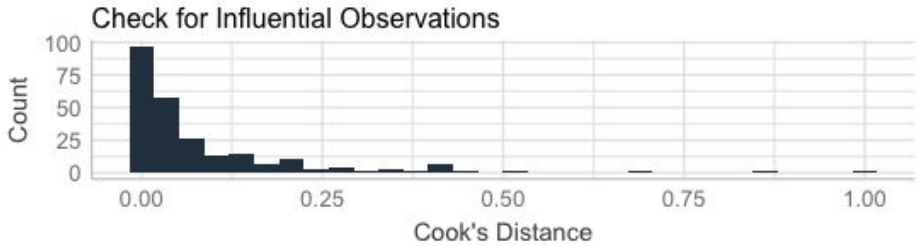












Durbin Watson Test

This tests for the non-independence of errors - our errors need to be independent (one of the assumptions of regression). This test needs the `car` package to be loaded.

```
> durbinWatsonTest(model2)
 lag Autocorrelation D-W Statistic p-value
   1     -0.03048832      2.055433   0.716
 Alternative hypothesis: rho != 0
```

A D-W value of 2 means that there is no autocorrelation in the sample - our calculated value is pretty close to that (p = 0.72) so we conclude our errors are independent of each other.

Stepwise Regression Based on AIC Improvement

Rather than building our regression model step by step manually, we can use the step function in R - it takes a starting model, and then uses forwards or backwards procedures (or a combination of both) to produce the best model.

Let's apply the procedure to `model0` and `model1` as our limits - we can specify the stepwise procedure with the parameter "direction":

```
> steplimitsboth <- step(model0,
                         scope = list(upper = model1),
                         direction = "both")
```

Stepwise Regression Based on AIC Improvement

```
> summary(steplimitsboth)
Call:
lm(formula = House_price ~ Crime + Population, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-27192.2  -6161.4   -555.2   6203.4  24061.0

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.736e+05  1.243e+04  13.973  < 2e-16 ***
Crime       -3.343e+02  1.147e+02  -2.915  0.00388 **
Population   6.662e-01  2.442e-01   2.729  0.00682 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9366 on 247 degrees of freedom
Multiple R-squared:  0.06084, Adjusted R-squared:  0.05323
F-statistic:     8 on 2 and 247 DF,  p-value: 0.0004301

> AIC(steplimitsboth)
[1] 5286.855
```

We can see the procedure has settled on the model with Crime and Population as the predictors.

AIC value is 5286.855.

In this case the stepwise model is the same as `model2` - which is what we arrived at manually.

Confidence Intervals of Parameter Estimates

We can also estimate the confidence intervals for each of our parameters using the `confint()` function - repeating the study, 95% of calculated confidence intervals will include the true value of the parameter.

```
> confint(model2)
                    2.5 %        97.5 %
(Intercept)  1.491596e+05 198110.856517
Population   1.853052e-01      1.147126
Crime       -5.602084e+02   -108.461481
```

Stepwise Regression Based on Adjusted R^2 Improvement

Use the `ols_step_forward()` function to work out the model with predictors entered on the basis of improvement via *p*-value and adjusted R^2. For this we need the package `olsrr`.

```
# Using ols_step_forward

> library(olsrr)
> pmodel <- ols_step_forward_p(model1)
> pmodel
```

```
> pmodel <- ols_step_forward_p(model1)
> pmodel

                            Selection Summary
--------------------------------------------------------------------------------
        Variable                    Adj.
Step     Entered      R-Square    R-Square     C(p)         AIC          RMSE
--------------------------------------------------------------------------------
   1    Crime           0.0325      0.0286    6.8913    5292.2791    9486.6020
   2    Population      0.0608      0.0532    1.4918    5286.8550    9365.6856
--------------------------------------------------------------------------------

> pmodel$model

Call:
lm(formula = paste(response, "~", paste(preds, collapse = " + ")),
    data = l)

Coefficients:
(Intercept)        Crime   Population
  1.736e+05   -3.343e+02    6.662e-01
```

The model determined by *p*-value improvement is also the one with the lowest AIC value - but this may not always be the case.

Model Diagnostics

In addition to the `check_model()` function, we can check the VIF values directly using the `vif()` function in the car package.

```
> vif(steplimitsboth)
     Crime Population
  1.000012   1.000012
```

As a rule of thumb VIF greater than 10 suggests a multicollinearity issue (although greater than 5 has been suggested too - more conservative).

For our case, we don’t have a collinearity problem as the VIF values are low.

The Model

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.736e+05  1.243e+04  13.973  < 2e-16 ***
Crime       -3.343e+02  1.147e+02  -2.915  0.00388 **
Population   6.662e-01  2.442e-01   2.729  0.00682 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Using these regression coefficients, we could write our regression equation as something like:

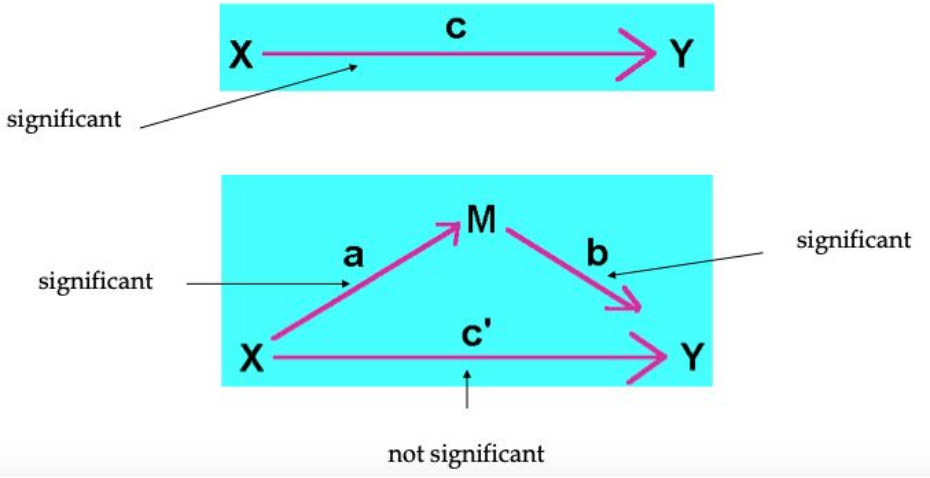
House price = 173,600 - 334.3(Crime) + 0.6662(Population) + residual

So, crime has a *negative* influence on house prices (more crime = lower prices) while population size has a *positive* influence on house prices (more people = higher house prices).

Mediation

At times, a variable's predicting power of an outcome is lost (complete mediation) or significantly reduced (partial mediation) when another predictor is introduced.

This added variable mediates the initial effect by its absence and cancels the effect by its presence.



Mediation Using Bootstrapping

You may have previously encountered Sobel Test for use in mediation. It can be quite conservative - bootstrapping and the `mediation` package a more powerful method to investigate mediation effect.

Bootstrapping is better for small samples.

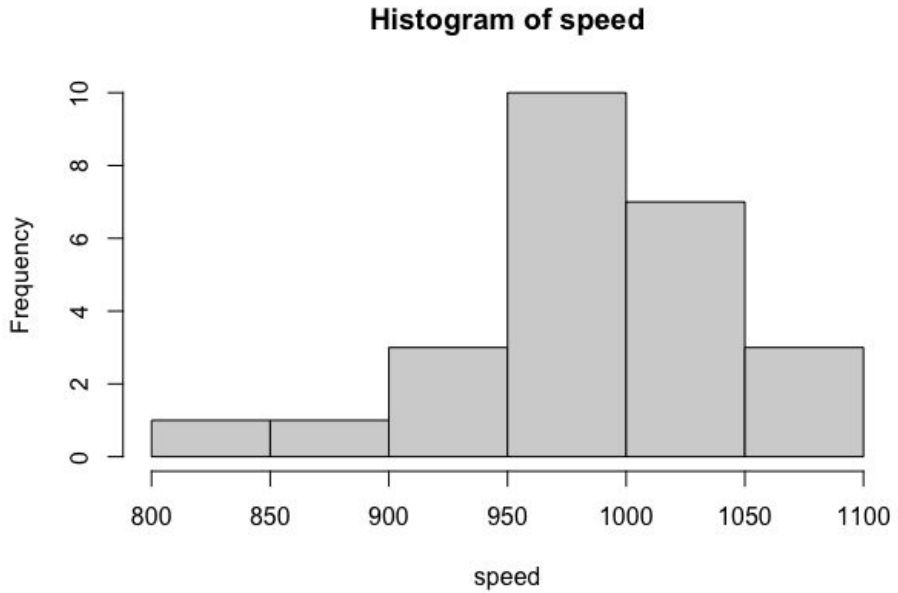
Imagine you have a sample of n measurements, you can sample this in many ways as long as you allow some values to appear more than once and others to be left out (sampling with replacement). Calculate the mean lots of times (one for each sampling - often as many as 10,000) and work out the CIs and other population parameter estimates. This is *bootstrapping*.

Example of Bootstrapping

Imagine we measure the speed of 25 military planes and plotted the data.

What might the underlying population of military planes we are sampling from look like?

We can use bootstrapping to investigate that.



Example of Bootstrapping

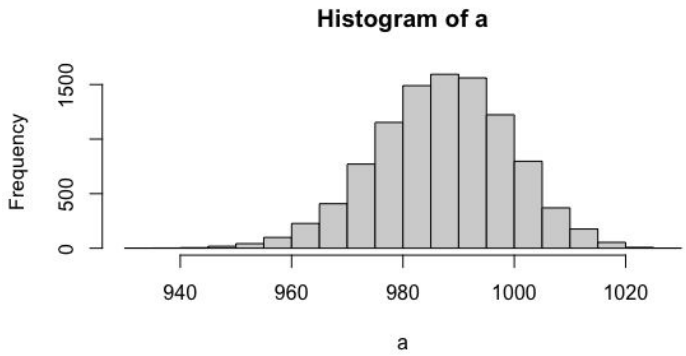
```
a <- numeric(10000)
for (i in 1:10000) {
  a[i] <- mean(sample(speed,replace = TRUE))
}
hist(a)
```

This bit of code creates 10,000 samples based on our 25 observations and plots a histogram of these 10,000 sample means.

We can use this to estimate population parameters such as mean and CI.

If we measure a plane with speed of less 950, how likely is that given our estimated population? We can work out the total number of planes with a speed of 950 (which is 18)…

The probably is therefore less than or equal to 18 in 10,000 (so p <= .0018)



Mediation in R

Imagine we are interested in whether the number of hours since dawn (X) affects the subjective ratings of wakefulness (Y) of 100 graduate students through the consumption of coffee (M).

The datafile is called `meddata.csv` and we will be using the package called `mediation`.

The following mediation and moderation content is from the interactive R book by Alexander Demos & Carlos Salas.

email: ademos@uic.edu

The full book with lots more content is available from here:

http://ademos.people.uic.edu/index.html

Mediation in R Using the Mediation Package

Based on Preacher & Hayes (2004) - more power than Sobel test - well suited for small sample sizes.

Need to build two models - one for direct effect of our predictors and one for the effect of our predictor and mediator.

```
> library(mediation)
> fitM <- lm(M ~ X, data = Meddata) #IV on M; Hours since dawn predicting coffee consumption
> fitY <- lm(Y ~ X + M, data = Meddata) #IV and M on DV; Hours since dawn and coffee predicting wakefulness
```

Then use the `mediate()` function in the mediation package to compare the two models - allows us to estimate the effect of the mediator.

```
> fitMed <- mediate(fitM, fitY, treat = "X", mediator = "M")
> summary(fitMed)

Causal Mediation Analysis

Quasi-Bayesian Confidence Intervals

               Estimate 95% CI Lower 95% CI Upper p-value
ACME             0.2808       0.1437         0.42  <2e-16 ***
ADE             -0.1133      -0.3116         0.09   0.258
Total Effect     0.1674       0.0208         0.34   0.028 *
Prop. Mediated   1.6428       0.5631         8.44   0.028 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Sample Size Used: 100


Simulations: 1000
```

By running this we get the Average Causal Mediation Effects (ACME), our Average Direct Effects (ADE), combined indirect and direct effects (Total Effect), and the ratio of these estimates (Prop. Mediated).

The ACME is the indirect effect of M (Total Effects - ADE) and the associated p-value value tells us if our mediation effect is significant.

We can bootstrap our data and fit a model based on our estimated population parameters (which is recommend over the default CI estimation method above).

```
> fitMedBoot <- mediate(fitM, fitY, boot = TRUE, sims = 10000, treat = "X", mediator = "M")
Running nonparametric bootstrap

> summary(fitMedBoot)

Causal Mediation Analysis

Nonparametric Bootstrap Confidence Intervals with the Percentile Method

               Estimate 95% CI Lower 95% CI Upper p-value
ACME            0.28078      0.14169         0.43  <2e-16 ***
ADE            -0.11179     -0.29809         0.10   0.278
Total Effect    0.16899     -0.00972         0.35   0.062 .
Prop. Mediated  1.66151     -3.43822        10.95   0.062 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Sample Size Used: 100


Simulations: 10000
```

We now see that the ACME is the only one that is significant - this tells us we have a significant mediator - with no direct effect of our predictor or combined effect of predictor and mediator when the mediator is taken into consideration.

Moderation

Moderation tests whether a variable (Z) affects the direction and/or strength of the relation between an IV (X) and a DV (Y).

In other words, moderation tests for interactions that affect when relationships between variables occur.

Imagine we are interested in whether the relationship between the number of hours of sleep (X) a student receives and the attention that they pay to this lecture (Y) is influenced by their consumption of coffee (Z).

Moderation

Our predictor variables are X and Z - good idea to centre them (i.e., mean of zero) using the scale function. It is important to centre both the moderator and predictor to make interpretation easier.

```
#Centering Data
Xc <- scale(X, center = TRUE, scale = FALSE)
#Centering IV, hours of sleep
Zc <- scale(Z, center = TRUE, scale = FALSE)
#Centering moderator, coffee consumption
```

```
> fitMod <- lm(Y ~ Xc + Zc + Xc:Zc) # Model interacts IV & moderator
> summary(fitMod)

Call:
lm(formula = Y ~ Xc + Zc + Xc:Zc)

Residuals:
    Min      1Q  Median      3Q     Max
-21.466  -8.972  -0.233   6.180  38.051

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 48.54443    1.17286  41.390  < 2e-16 ***
Xc           5.20812    0.34870  14.936  < 2e-16 ***
Zc           1.10443    0.15537   7.108 2.08e-10 ***
Xc:Zc        0.23384    0.04134   5.656 1.59e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.65 on 96 degrees of freedom
Multiple R-squared:  0.7661,  Adjusted R-squared:  0.7587
F-statistic: 104.8 on 3 and 96 DF,  p-value: < 2.2e-16
```

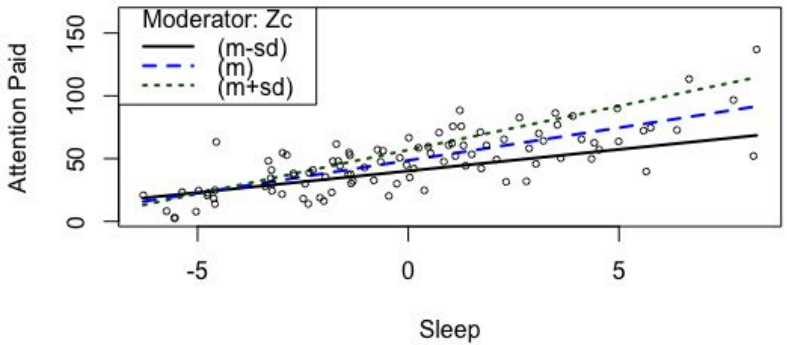
We have an effect of moderation (i.e., an interaction effect) between Xc and Zc.

We can use the `plotSlopes()` function in the `rockchalk` package to plot the slopes (1 SD above and 1 SD below the mean) of the moderating effect.

The plot below shows that those who drank less coffee (the black solid line) paid more attention with the more sleep they got last night, but paid less attention overall than average (the blue dotted line).

Those who drank more coffee (the green dotted line) paid more attention when they slept more as well and paid more attention than average.

The difference in the slopes for those who drank more or less coffee shows that coffee consumption moderates the relationship between hours of sleep and attention paid.




```
plotSlopes(fitMod, plotx = "Xc", modx
= "Zc", xlab = "Sleep", ylab =
"Attention Paid", modxVals =
"std.dev")
```