

# Workshop 2- General Linear Model - Regression

Andrew Stewart

Andrew.Stewart@manchester.ac.uk



@ajstewart\_lang



<https://github.com/ajstewartlang>

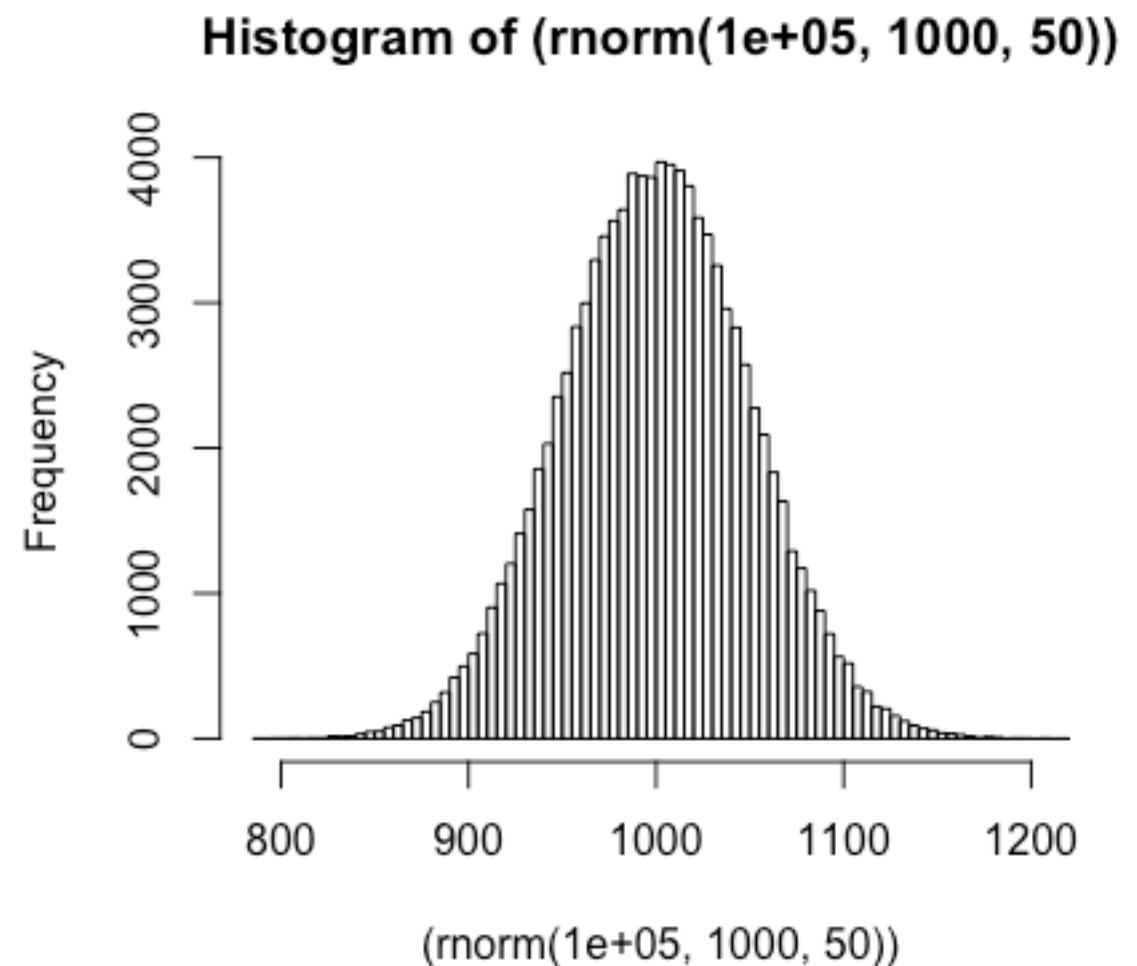
Workshop	Topic
1	Reproducibility and R
2	General Linear Model (Regression)
3	General Linear Model (ANOVA)
4	Mixed Models
5	Data Simulation and Advanced Data Visualisation
6	Reproducible Computational Environments and Presentations

### **Assignment**

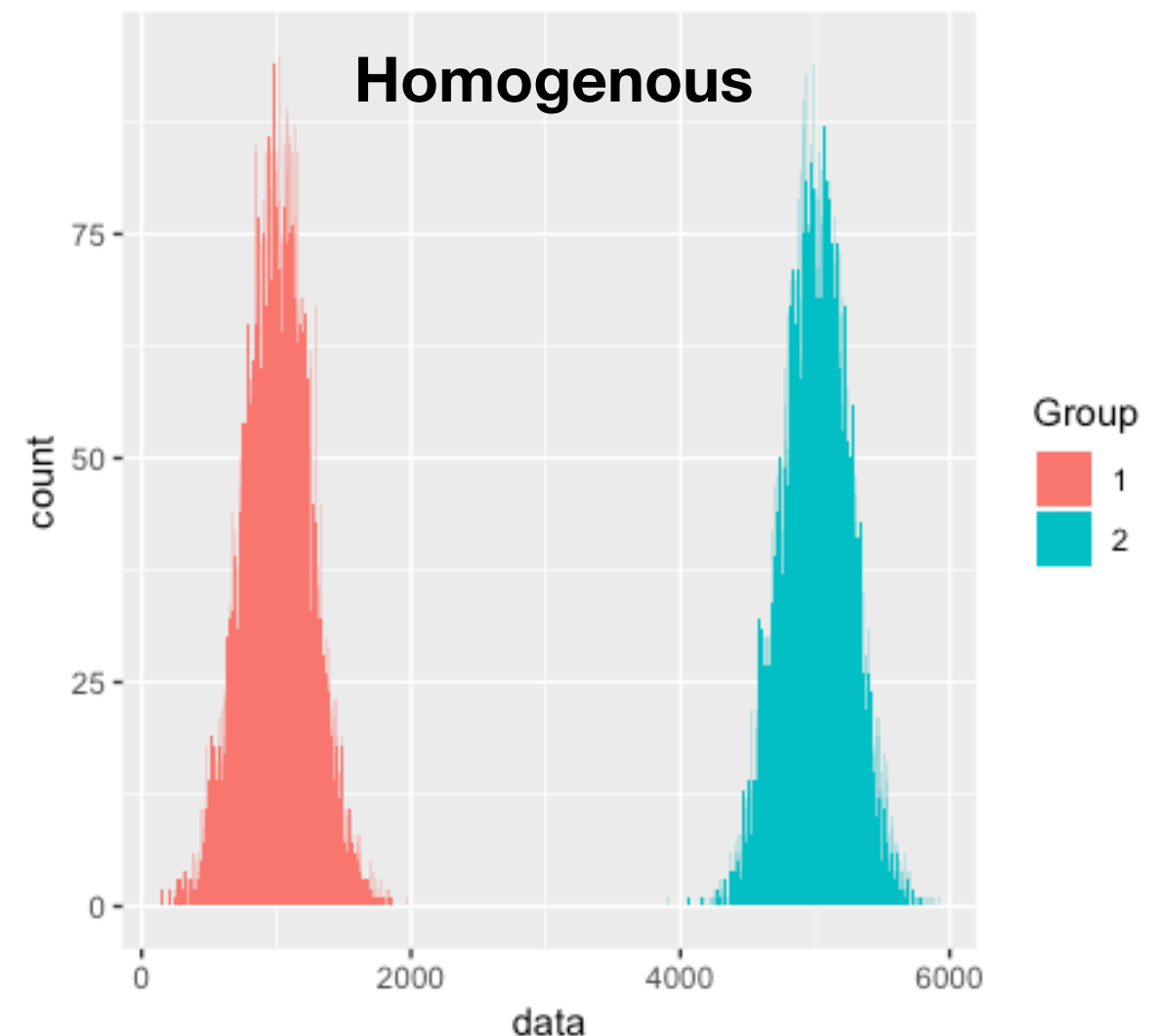
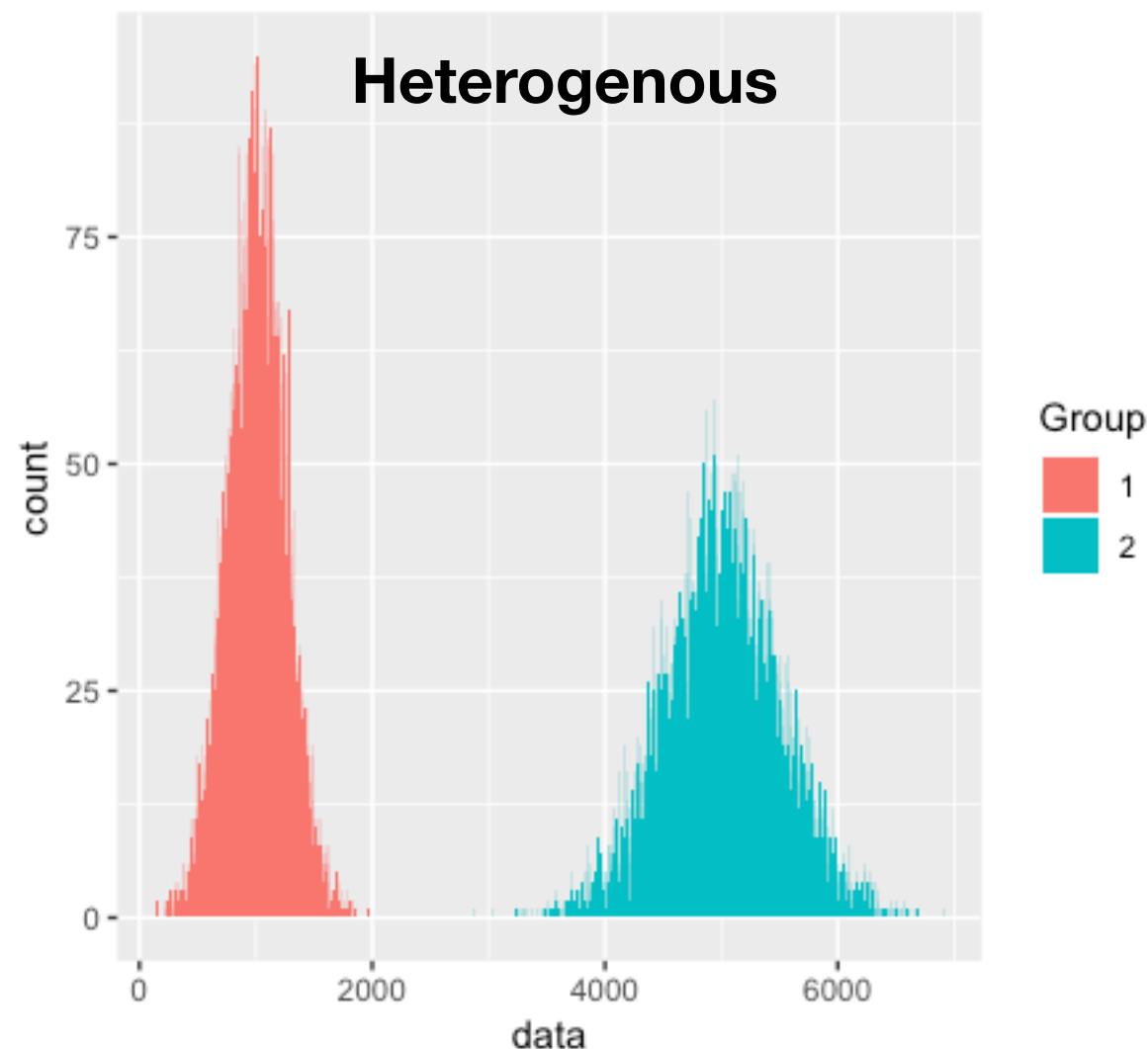
Assignment to be completed by Semester 1 exam period.

# A Brief Reminder of the Assumptions of Parametric Tests

- Assumption 1 - the data are conditionally normally distributed - in practical terms, this means the *residuals* need to be normally distributed (although t-tests require the data to be normal).



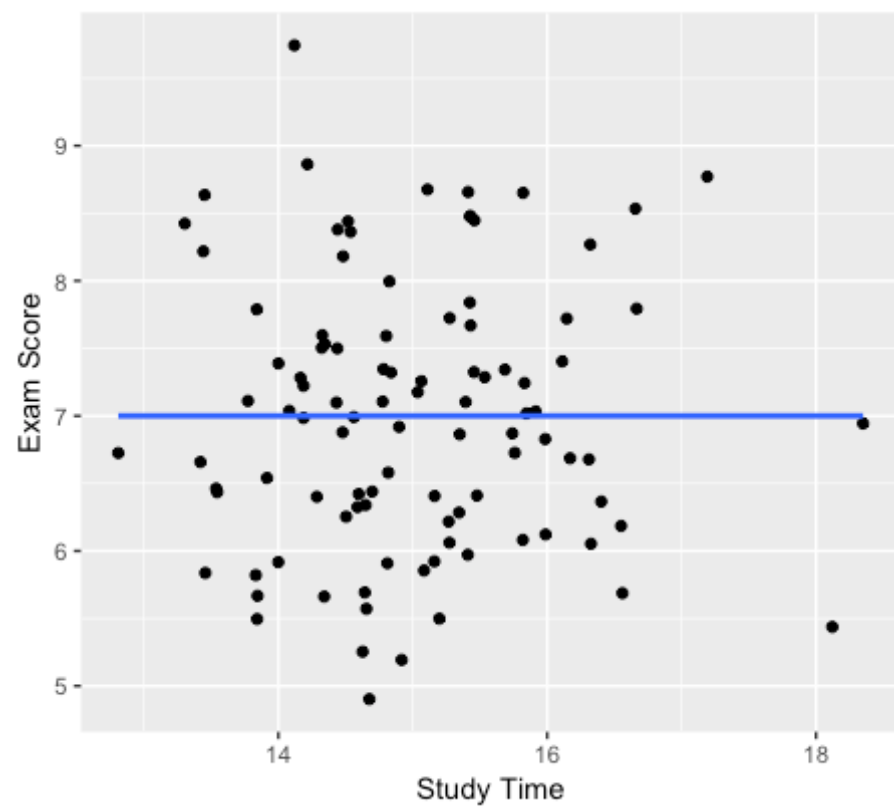
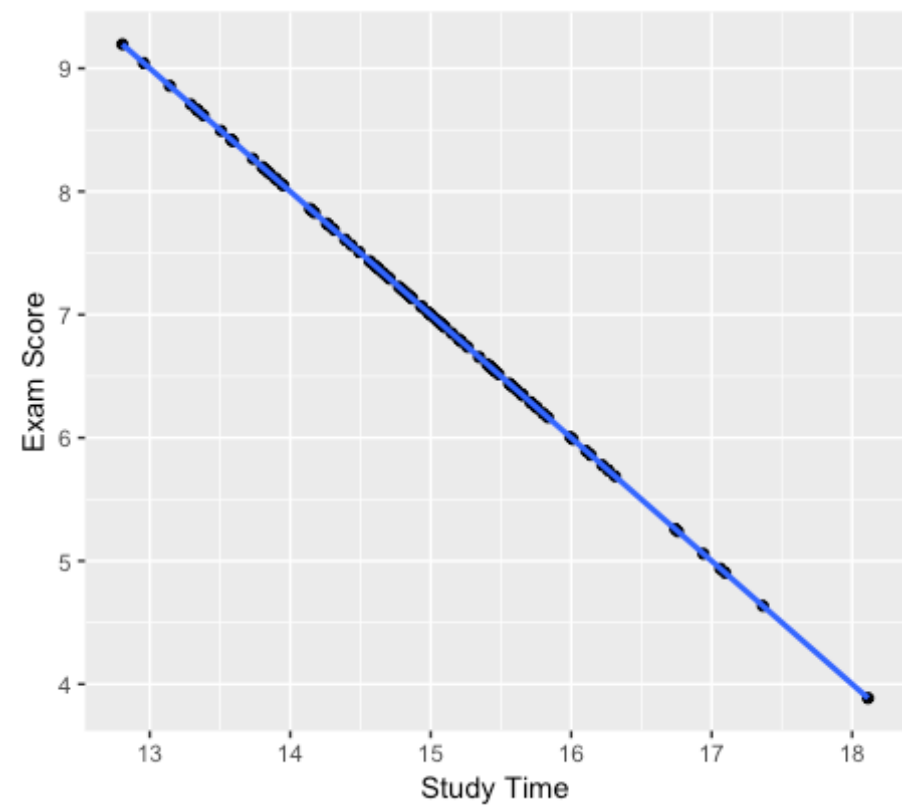
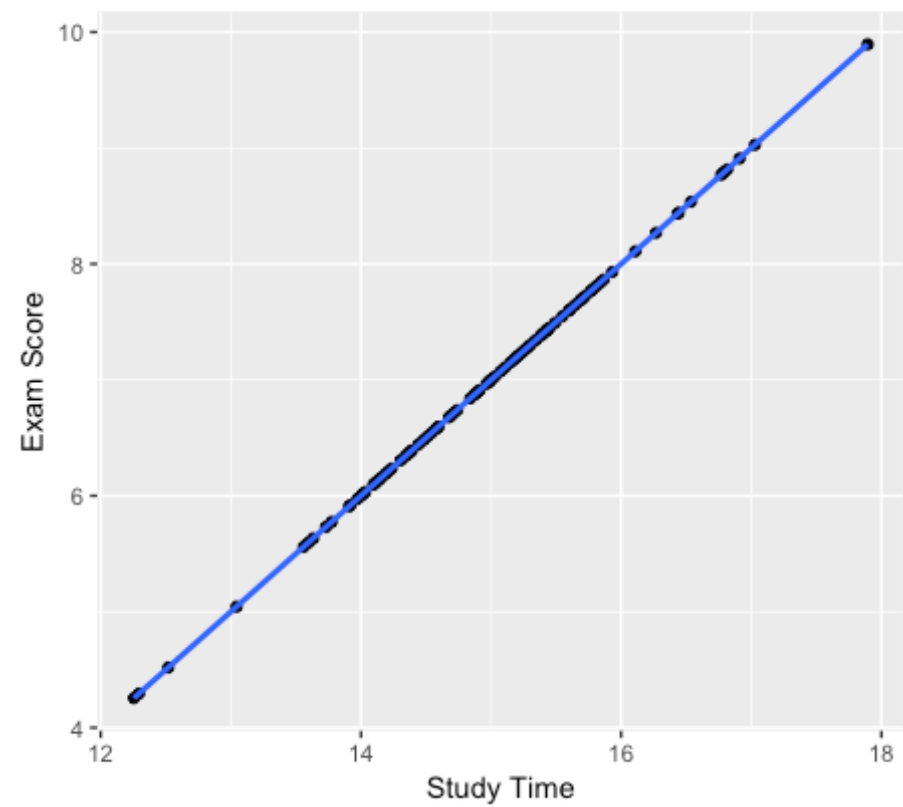
- Assumption 2 – Homogeneity of variance – the variances should not change systematically throughout the data. In designs where you test several groups of participants this means that the variances of each group should be equivalent.
- Levene's Test for equality of variance. If it is non-significant, then it means that the variances are equivalent (i.e., we have homogeneity of variance).



- Assumption 3 – Interval data – data should be measured on an interval scale. In other words, the distance between two adjacent points should be the same as the distance between any other two adjacent points. R can't tell you this – you need to determine it by yourself. Reaction time is a good example of interval data.
- Assumption 4 – Independence. The data from one participant does not affect the data from another (i.e., they are independent). In repeated measures designs, we expect the scores in the experimental conditions to be independent between participants.

# Remember correlation?

- Sometimes we want to know whether there's a relationship between two variables –
  - time spent studying statistics and performance in exam.
- They could be *positively* correlated, *negatively* correlated or *uncorrelated*.



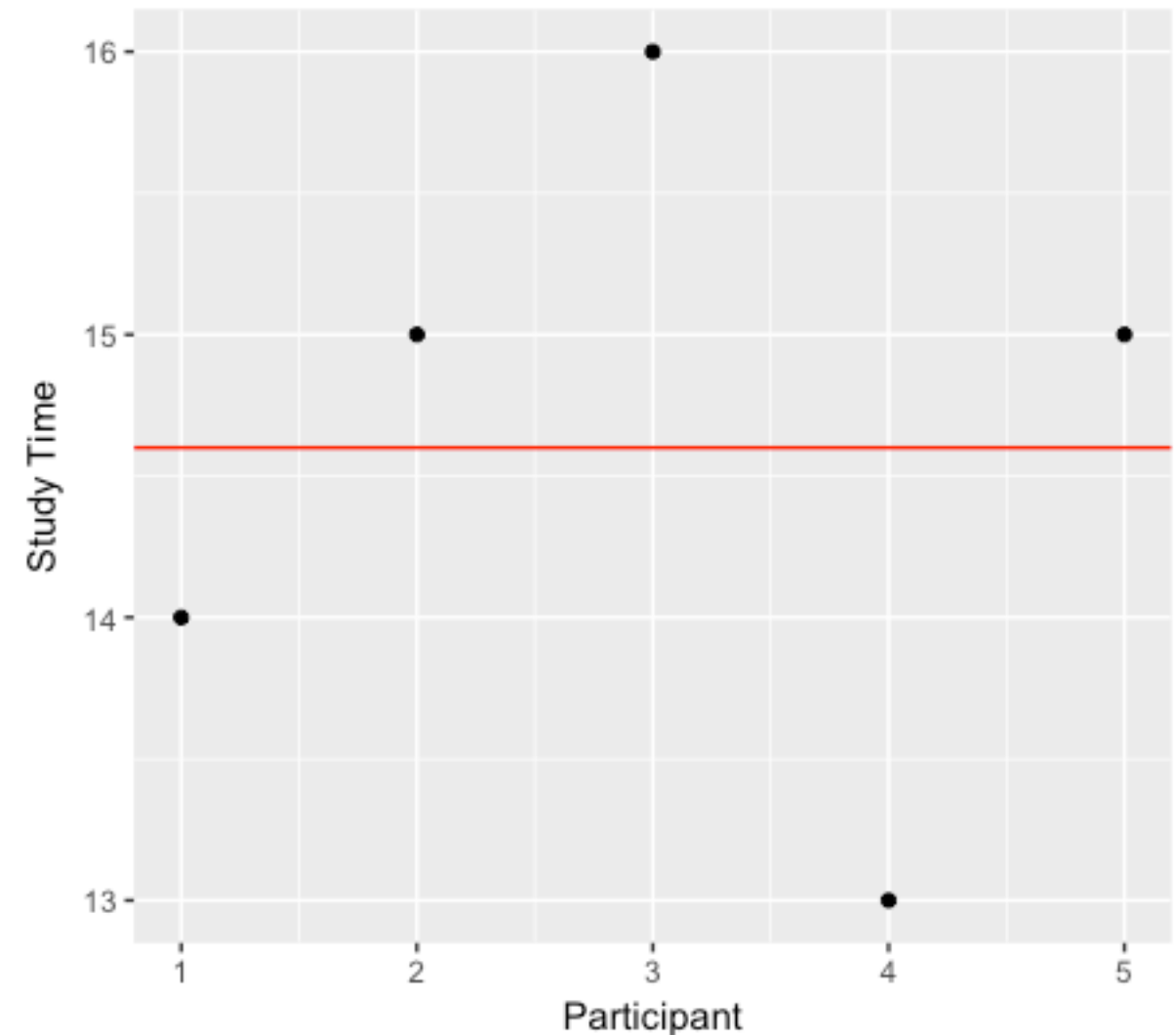
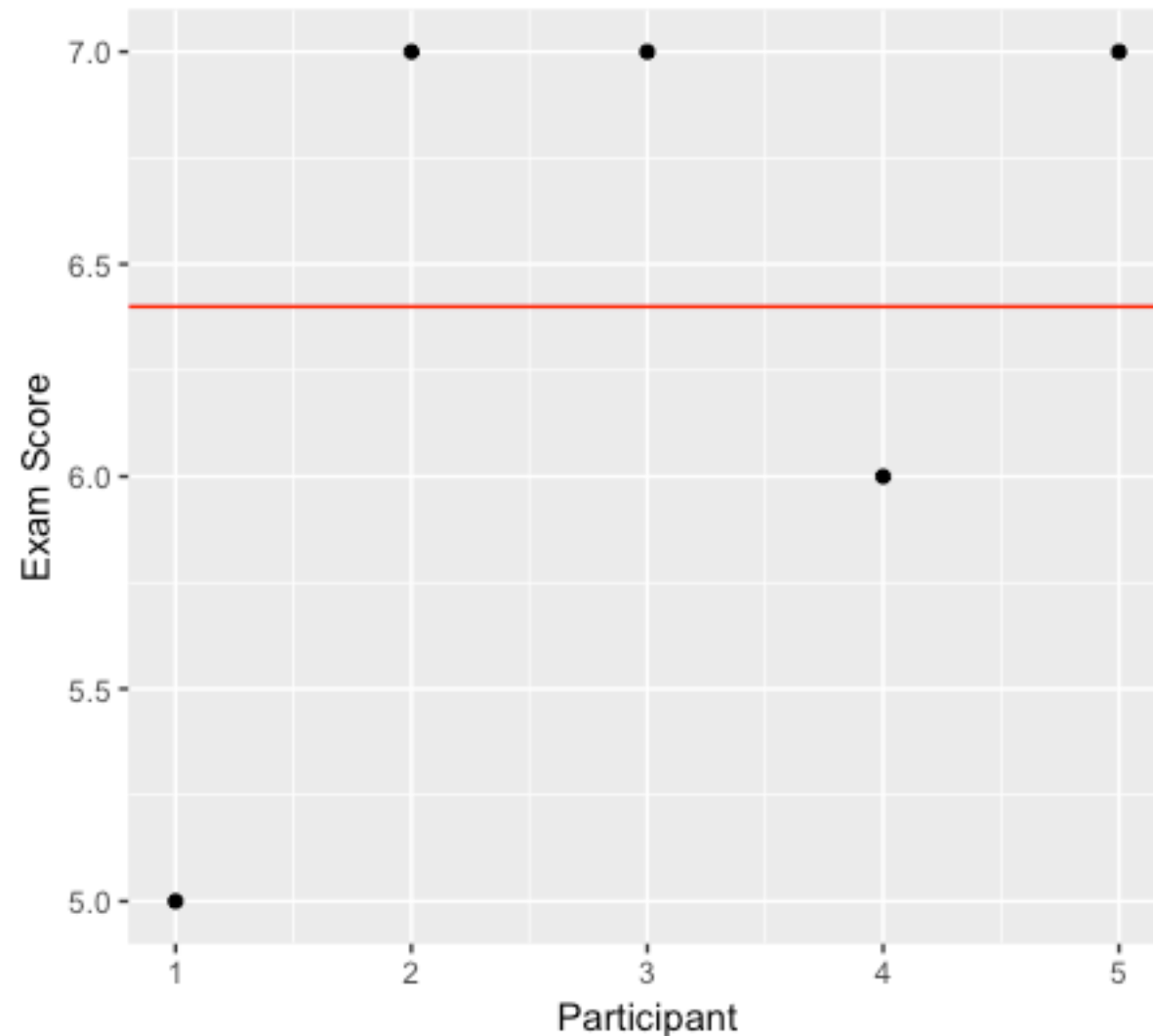
# Covariance

- Remember variance?
- It's the measure of the average amount by which data associated with a variable vary from the mean of that variable...

$$= \frac{\sum (x_i - \bar{x}) (x_i - \bar{x})}{N - 1}$$

- If two variables *covary*, then when one variable deviates from the mean, we expect the other variable to deviate from its mean in a similar way.





The red horizontal lines represent the mean for each variable - if a participant is below the mean on one variable, notice that they are also below the mean for the other variable - this suggests the two variables co-vary.

- For participants 2, 3 and 5, their scores on each variable are all below the respective means for each variable, for participants 1 and 4 their scores are all above the respective means for each variable.
- To formalise this, we can calculate the average combined differences.....

$$\text{cov}(x,y) = \frac{\sum (x_i - \bar{x}) (y_i - \bar{y})}{N - 1}$$

- For our example:

Participant	Study Time (X)	Exam Score (Y)	Mean X	Mean Y	X - Mean X	Y - Mean Y	(X - Mean X) * (Y - Mean Y)
1	14	5	14.6	6.2	-0.6	-1.2	0.72
2	15	7	14.6	6.2	0.4	0.8	0.32
3	16	7	14.6	6.2	1.4	0.8	1.12
4	13	6	14.6	6.2	-1.6	-0.2	0.32
5	15	7	14.6	6.2	0.4	0.8	0.32

**$\Sigma = 2.8$**

$$\text{Cov}(x,y) = 2.8/N-1 = 2.8/4 = 0.7$$

- Now, one problem with covariance as we've calculated it is that the score we end up with depends on the measurement scales associated with our variables.
- In other words, the covariance value isn't standardised.
- We can divide any value by the standard deviation and that will give us the distance from the mean in standard deviation units....

- We can divide our covariance value by the standard deviations of our two variables (actually standard deviation of x multiplied by standard deviation of y) – in other words:

$$= \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N - 1 s_x s_y}$$

- This is called the *Pearson product-moment correlation coefficient* and ranges from -1 (perfect negative correlation) to +1 (perfect positive correlation) with 0 meaning no correlation at all.

- SD of Study Time (X) = 1.140175
- SD of Exam Score (Y) = 0.8944272

$$\text{Pearson's } R = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N - 1 s_x s_y}$$

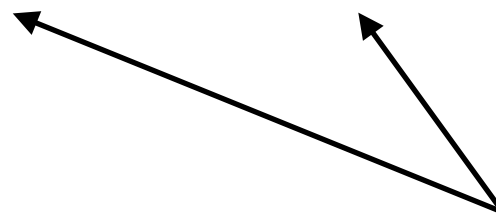
$$\begin{aligned} \text{Pearson's } R &= \frac{2.8}{4 \times 1.14 \times 0.89} \\ &= 0.69 \end{aligned}$$

- In R, you need to install the `Hmisc` package first, and then load it:

```
> library(Hmisc) # Needed for correlation
```

- Our data frame is called “covary” and looks like this:

```
> covary
# A tibble: 5 x 3
  Participant Study_time Exam_score
      <dbl>      <dbl>      <dbl>
1           1         14           5
2           2         15           7
3           3         16           7
4           4         13           6
5           5         15           7
```



- To calculate Pearson’s R for these two variables we type:

```
> rcorr(covary$Study_time, covary$Exam_score)
```

```

> rcorr(covary$Study_time, covary$Exam_score)
      x      y
x 1.00 0.69
y 0.69 1.00

n= 5

P
      x      y
x      0.2006
y 0.2006

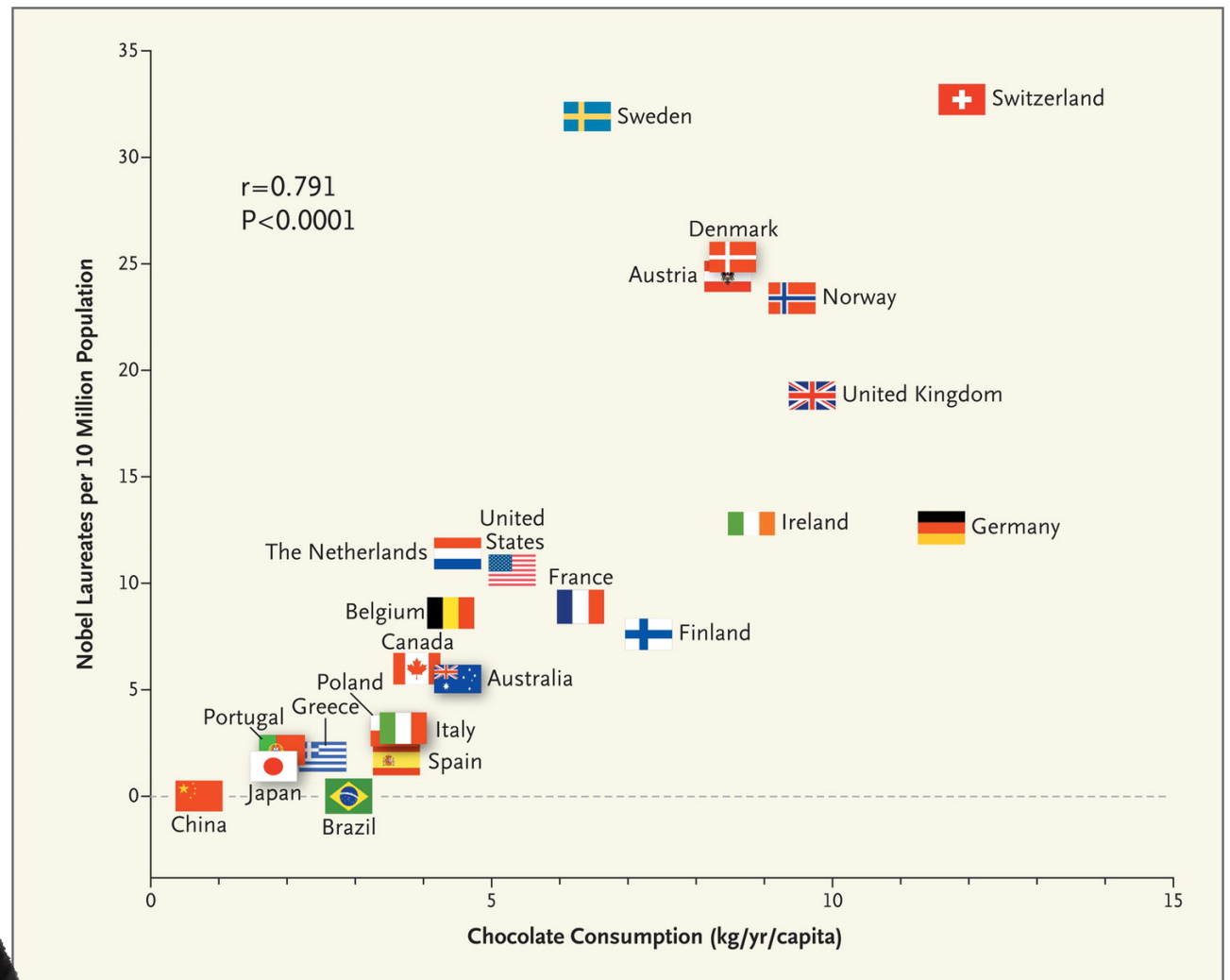
```

- The Pearson's r value is 0.69 with  $p = 0.20$



# Correlation is *not* Causation

There is a high correlation ( $r = 0.791$ ) between chocolate consumption in a country and the number of Nobel Prize winners in that country...Why do you think this is?



# Correlation is *not* Causation

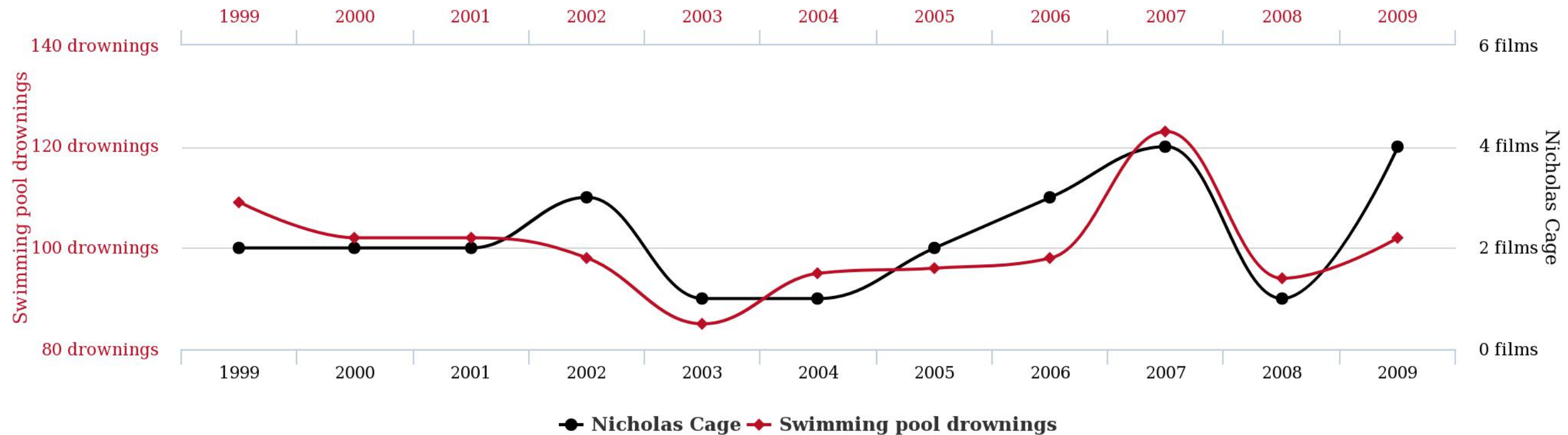
- When interpreting correlation data one common pitfall is to assume that the score on one variable *causes* a particular score on the other. This is wrong!
- Very often, common sense would suggest causation – e.g., time spent studying improves exam score. Again, you cannot make any claim about causation from correlation.
- There may be a third variable that we don't know about – in this case, maybe a positive attitude to studying.
- Additionally, spurious correlations can be found all over the place...

# Correlation is *not* Causation

**Number of people who drowned by falling into a pool**

correlates with

**Films Nicolas Cage appeared in**

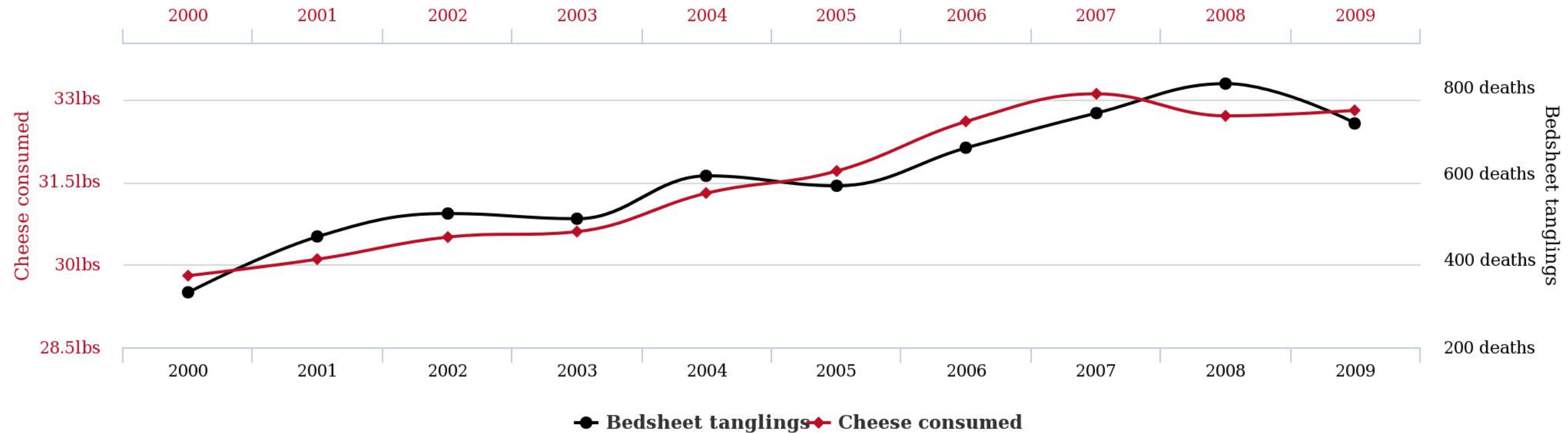


# Correlation is *not* Causation

**Per capita cheese consumption**

correlates with

**Number of people who died by becoming tangled in their bedsheets**

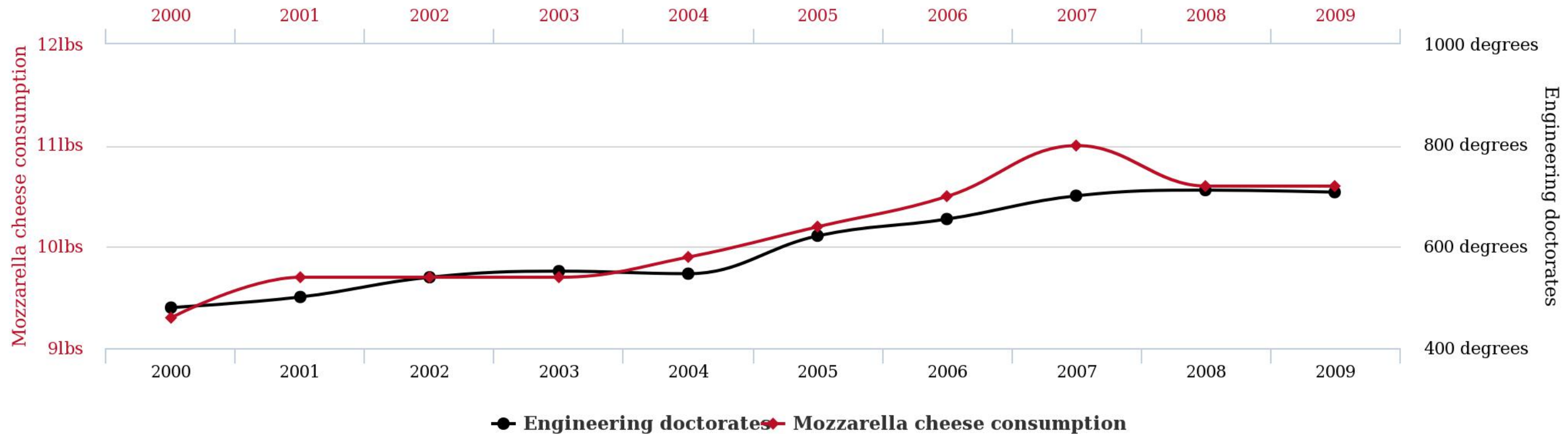


# Correlation is *not* Causation

**Per capita consumption of mozzarella cheese**

correlates with

**Civil engineering doctorates awarded**



# R squared – How much variance in one variable can be explained by the other?

- Simply square Pearson's  $r$  to get  $r$  squared.
- If we multiple this value by 100, that will be the % of variance explained in one variable by the other.
- For our example on time spent studying and exam score,  $r$  squared = 0.4761 as  $r = 0.69$
- This means that about 48% of the variance in exam score is explained by time spent studying. It may not be statistically significant, but you might think it is still meaningful.

# Regression

- Regression is where we want to predict the value of one variable (called our Outcome variable) on the basis of the value of one or more predictor variables.
- Simple regression is when we have one predictor, multiple regression is when we have more than one...
- Most commonly used regression type is OLS (ordinary least squares) which works by minimising the distance (deviation) between the observed data and the linear model.

# Statistical Models

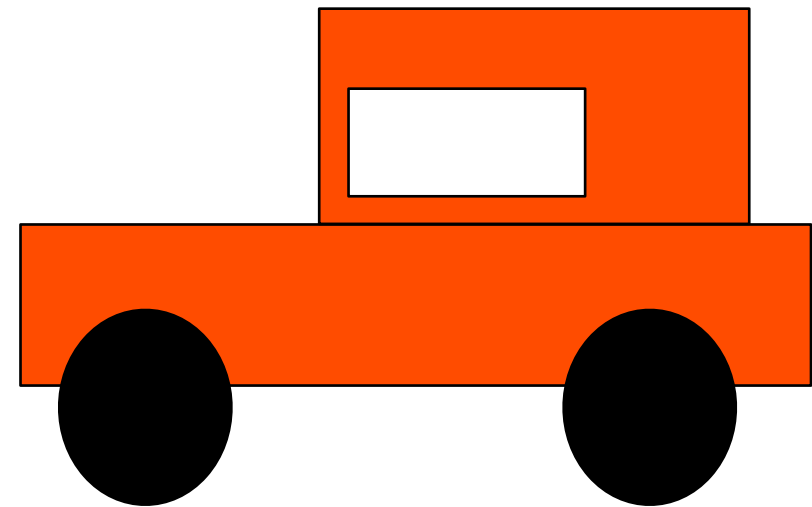
- Most of what we do in applying statistics in Psychology is model building. We build a statistical model and test whether it is a good fit for our data - in other words, whether it describes our data well.
- All models are an approximation of reality, and some are better than others...
- Or to paraphrase the statistician George Box, all models are wrong but some are useful...



Real data



Model 1



Model 2

- So how do we tell if a particular statistical model is a good fit to our data?
- We can look at the extent to which our data deviate from a particular model (where deviation = error)...

Real data



=

Model 1



+ Error

Real data



=

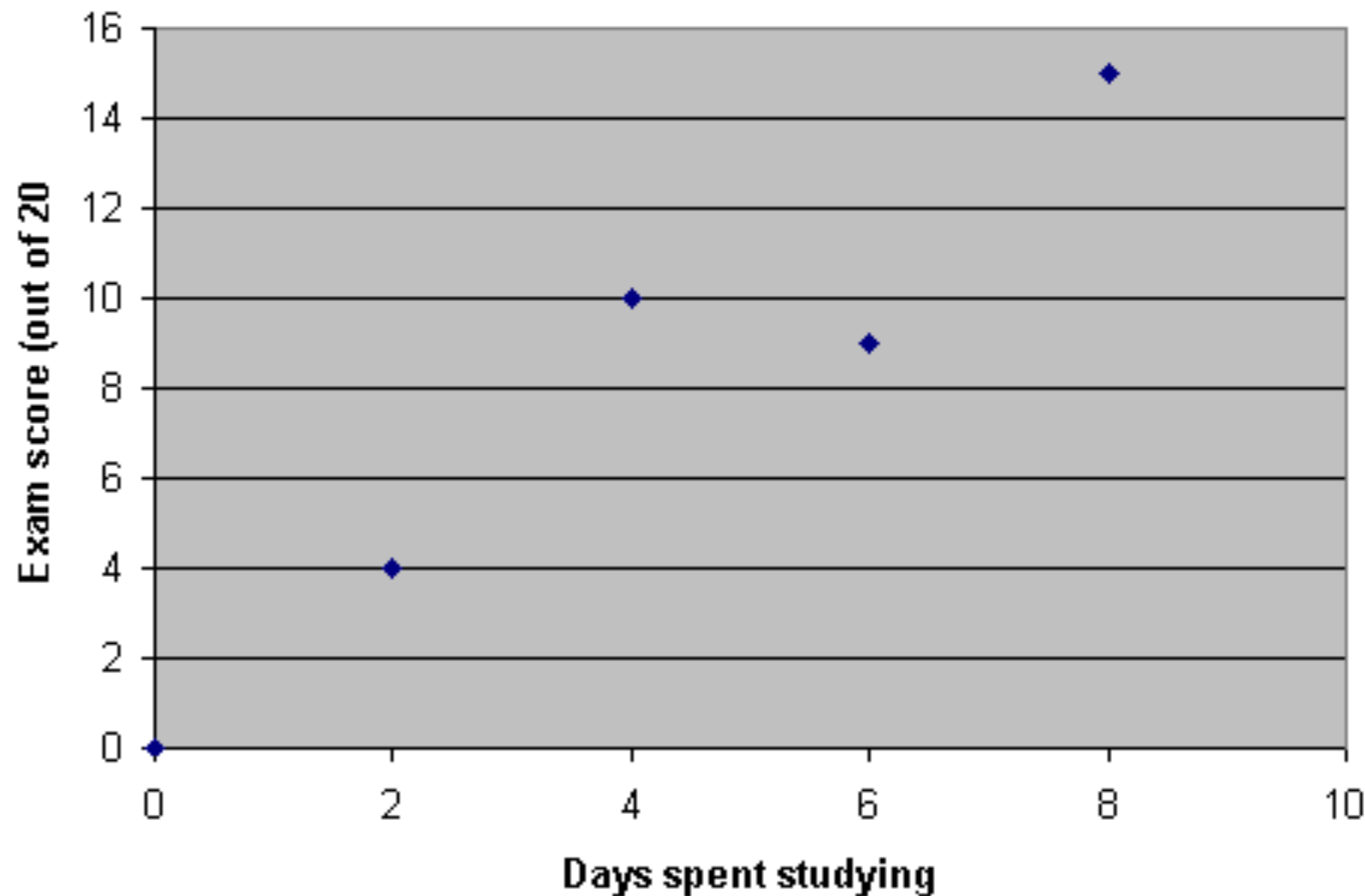
Model 2



+ Error

- We want to select the model which has the smallest error (aka model residuals)...

# Regression



We can plot data on exam performance and days spent studying.

Wouldn't it be helpful if we could draw a straight line such that if we know the value on one axis (x say), we could predict the value on the other (y say) ?

# Plotting a straight line

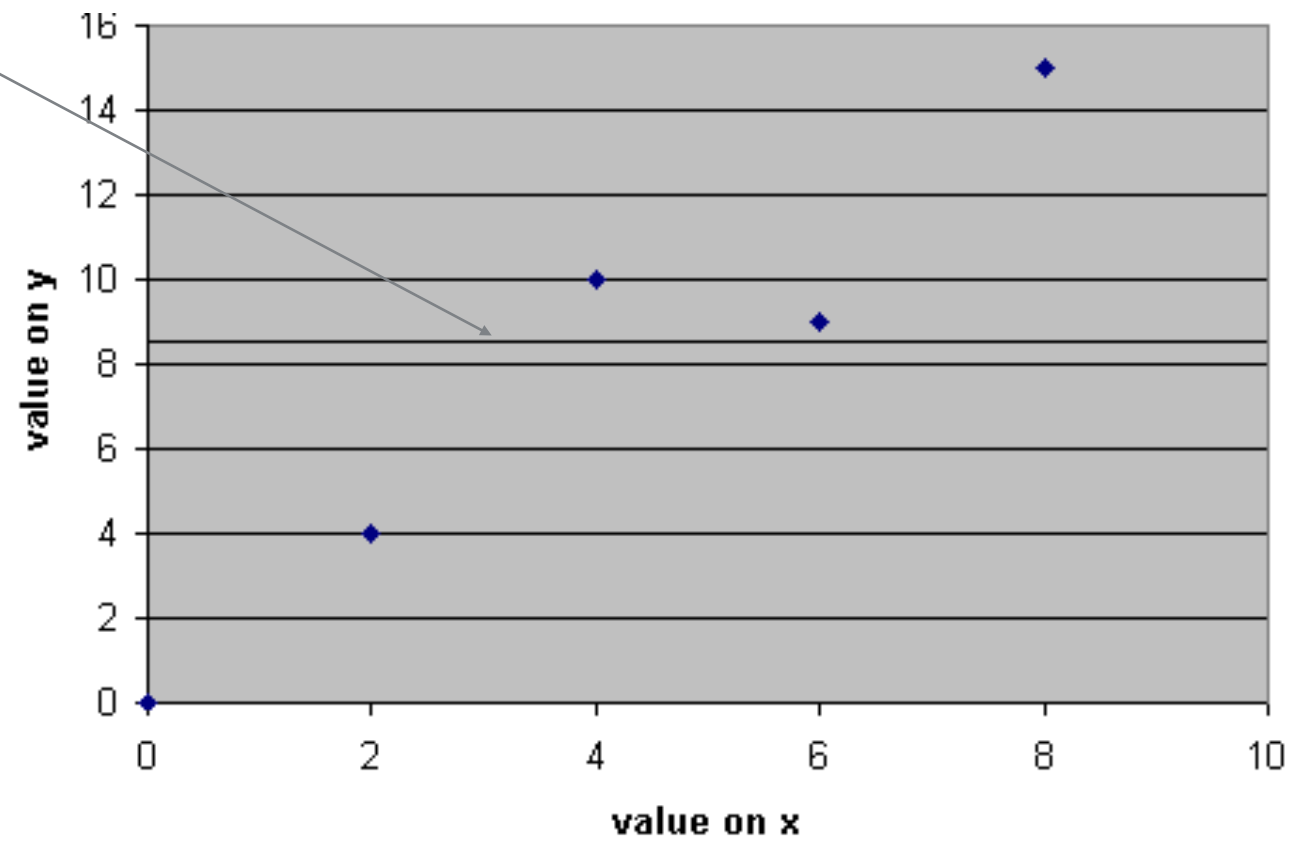
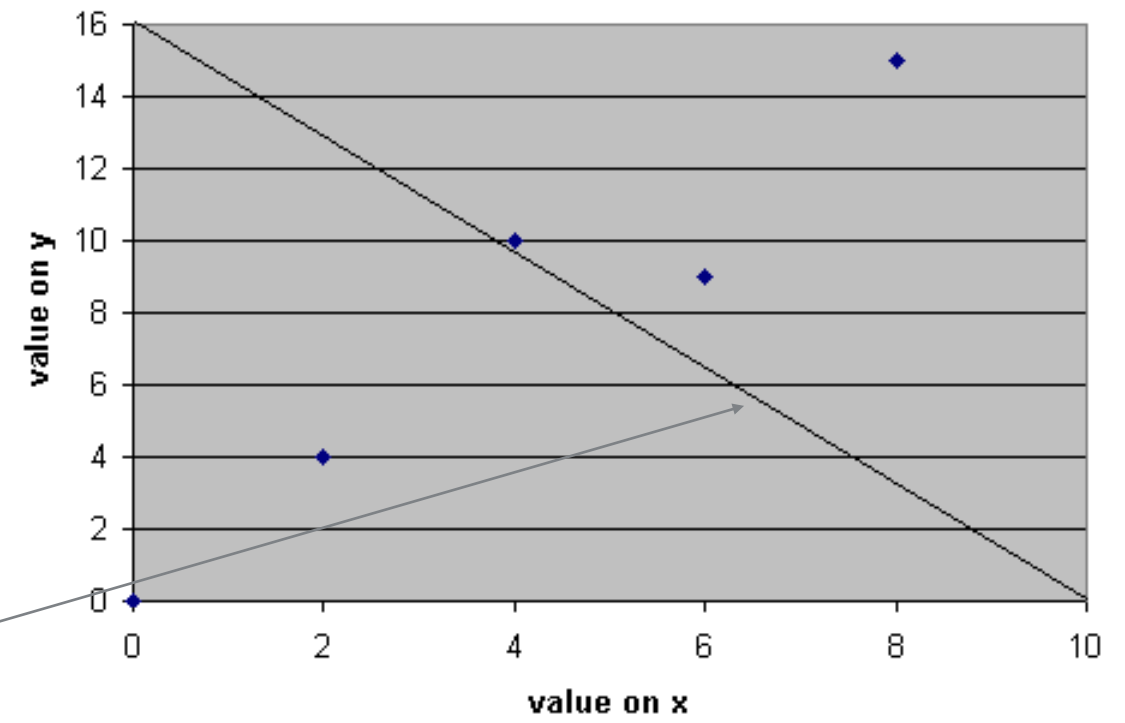
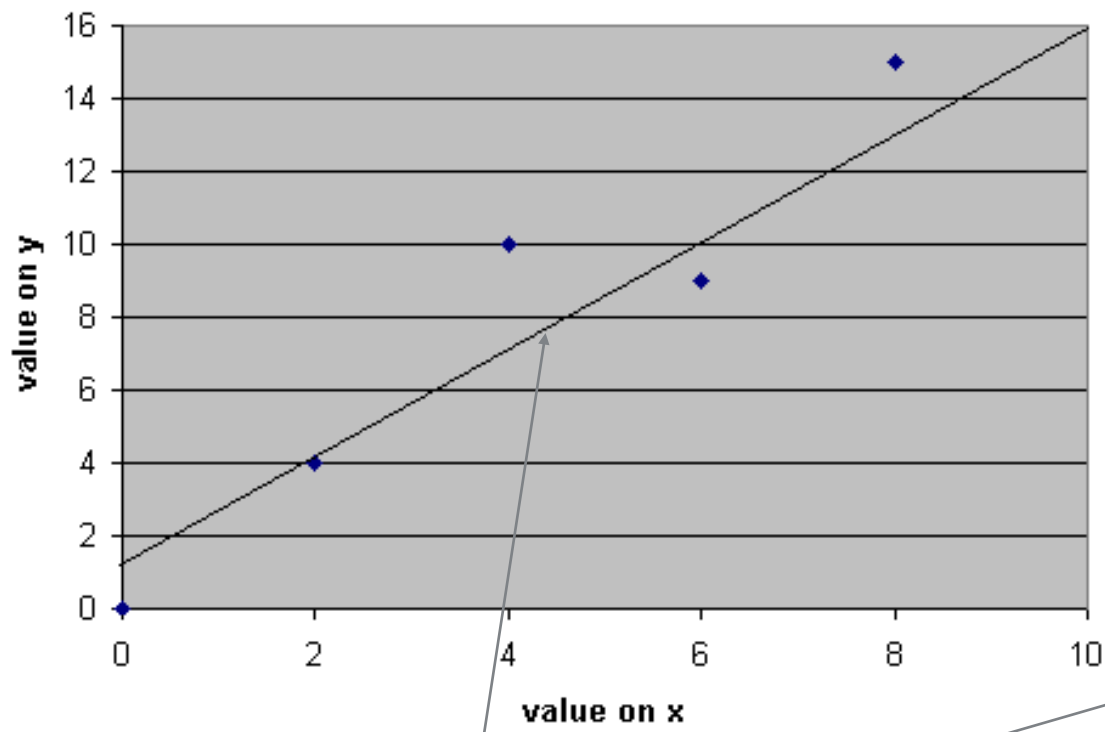
- For any data plots such as on the previous slide, when we have one predictor (x) we could plot many straight lines.

$$y = \beta x_i + \beta_0 + \text{residual}_i$$

$\beta$  = gradient of the line

$\beta_0$  = intercept (when  $x=0$ )

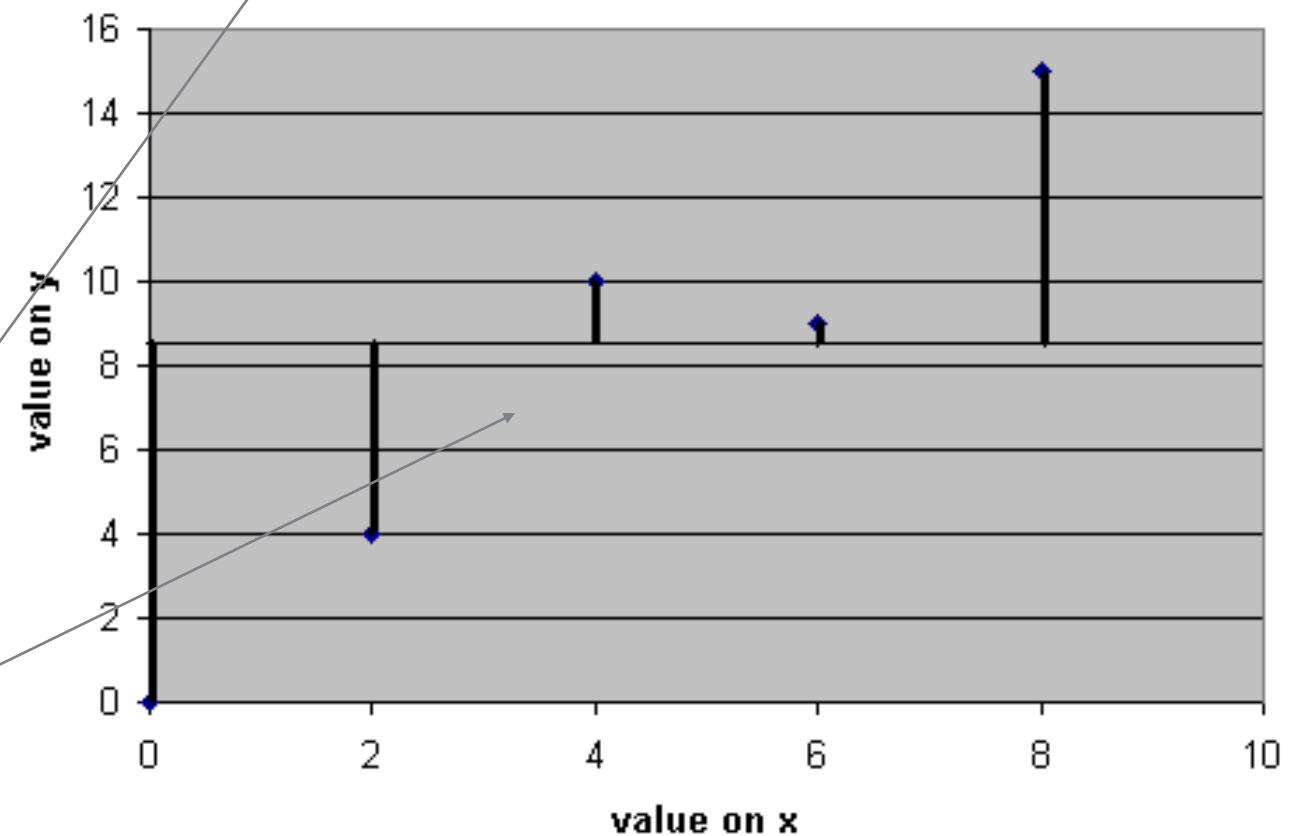
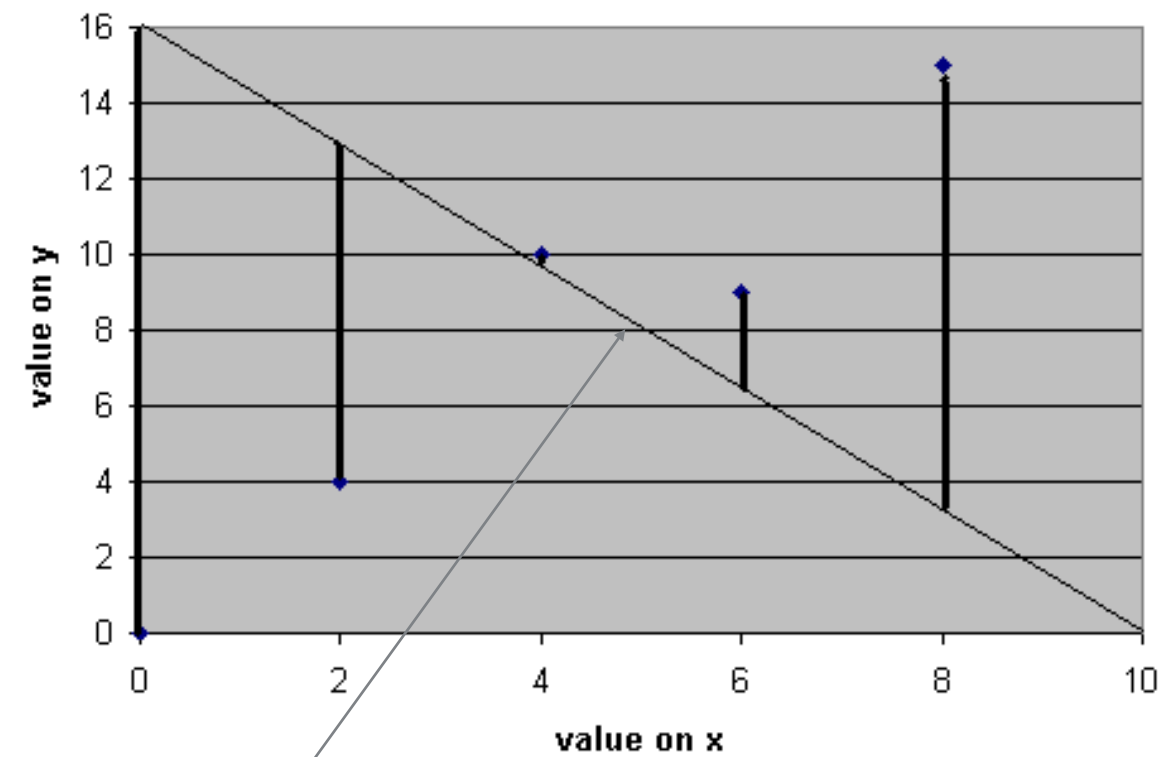
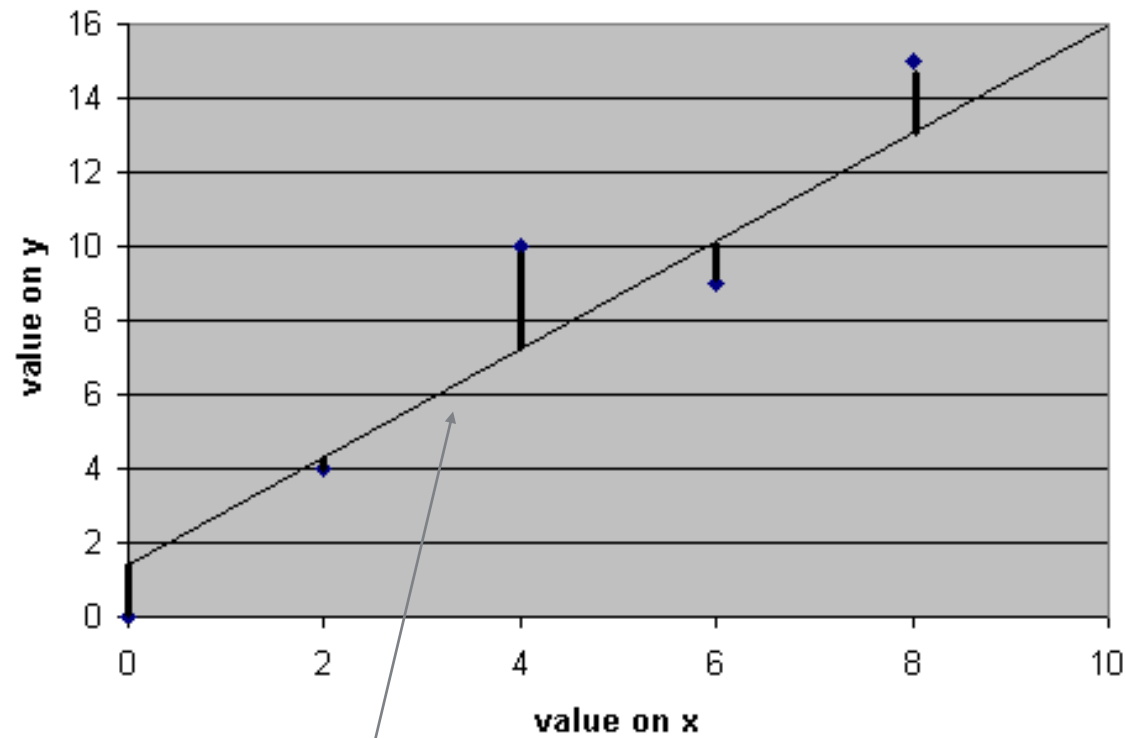
$\text{residual}_i$  = difference between predicted score and actual score for participant  $i$



Which line  
seems the  
best fit ?

# Determining the best line

- For any line, we can calculate what's known as the Least Squares.
- The Least Squares method in regression provides us with a line that results in the least differences between the values predicted by the line and the data themselves....
- So, for the three possible lines we just looked at....



We can see that this line seems to be the best fit as it leads to the least error between the predicted data (the line) and our observed data (the points).

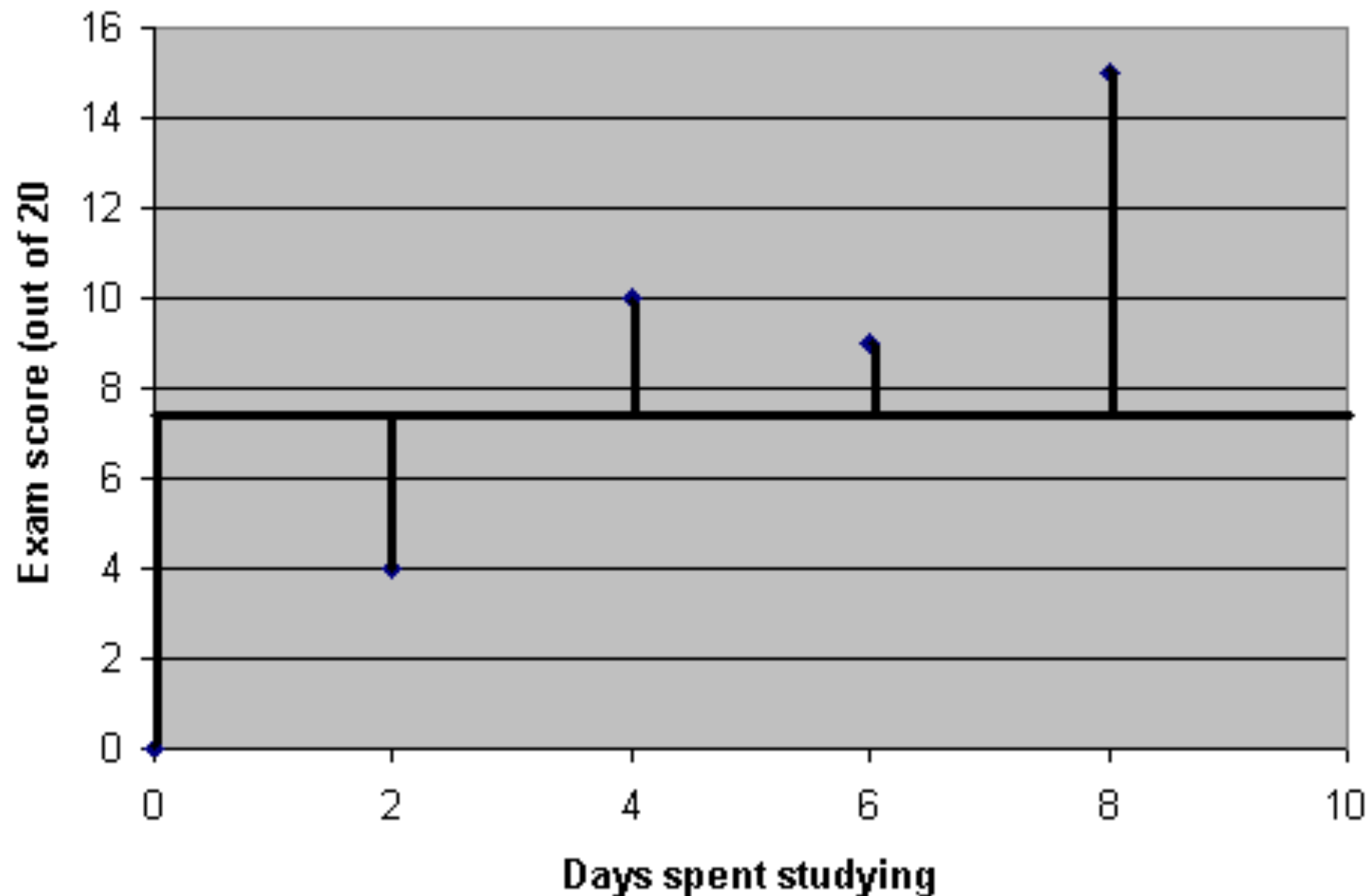
These two lines aren't much good as they lead to a lot of error between predicted and observed data.

# How do we determine how good a fit our line is ?

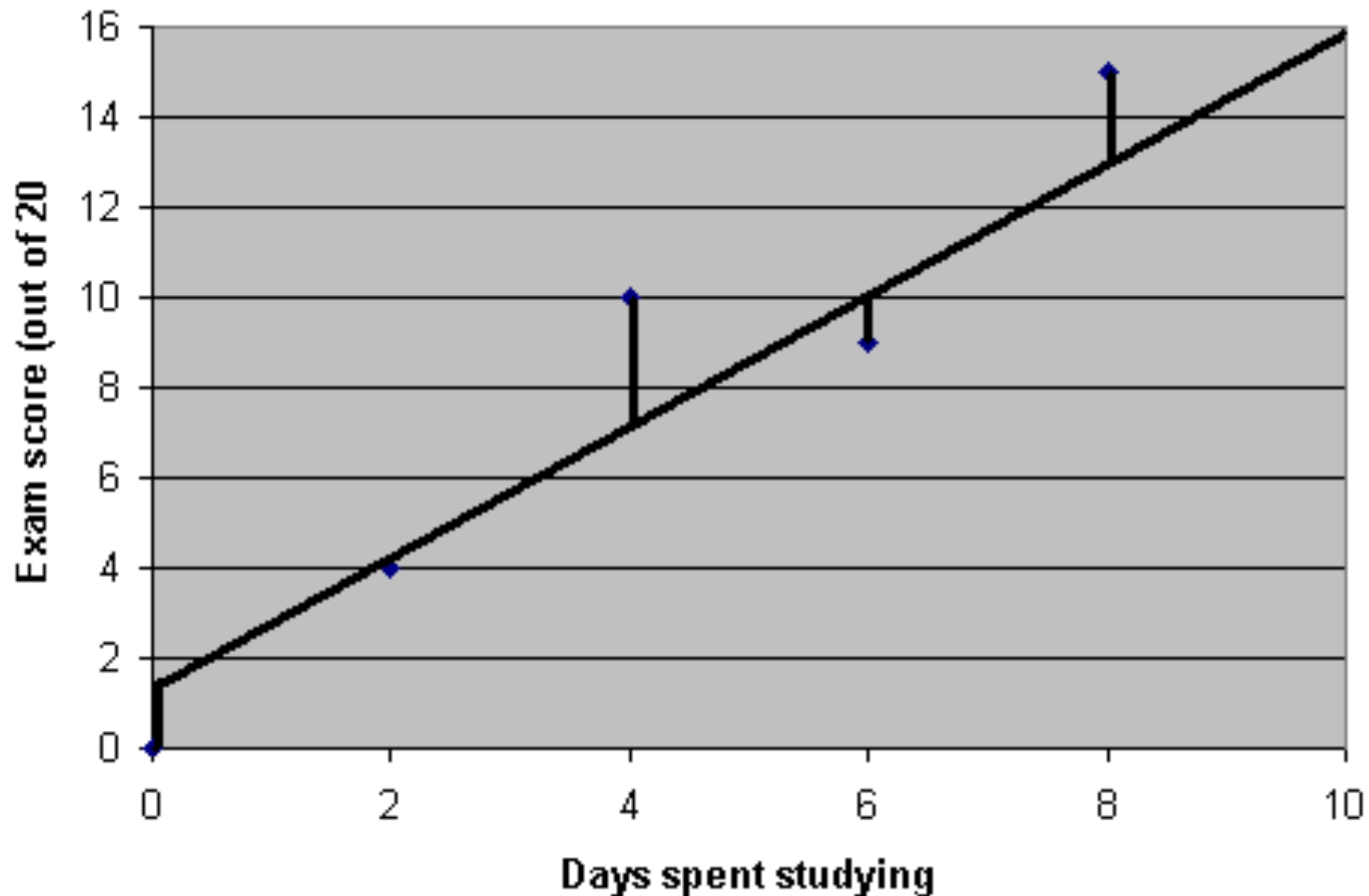
- We could work out by how much each observed value differs from the mean of  $y$ .
- We could work out by how much each observed value differs from the regression line.
- We could work out by how much the mean value of  $y$  differs from the regression line (for different values of  $x$ ).



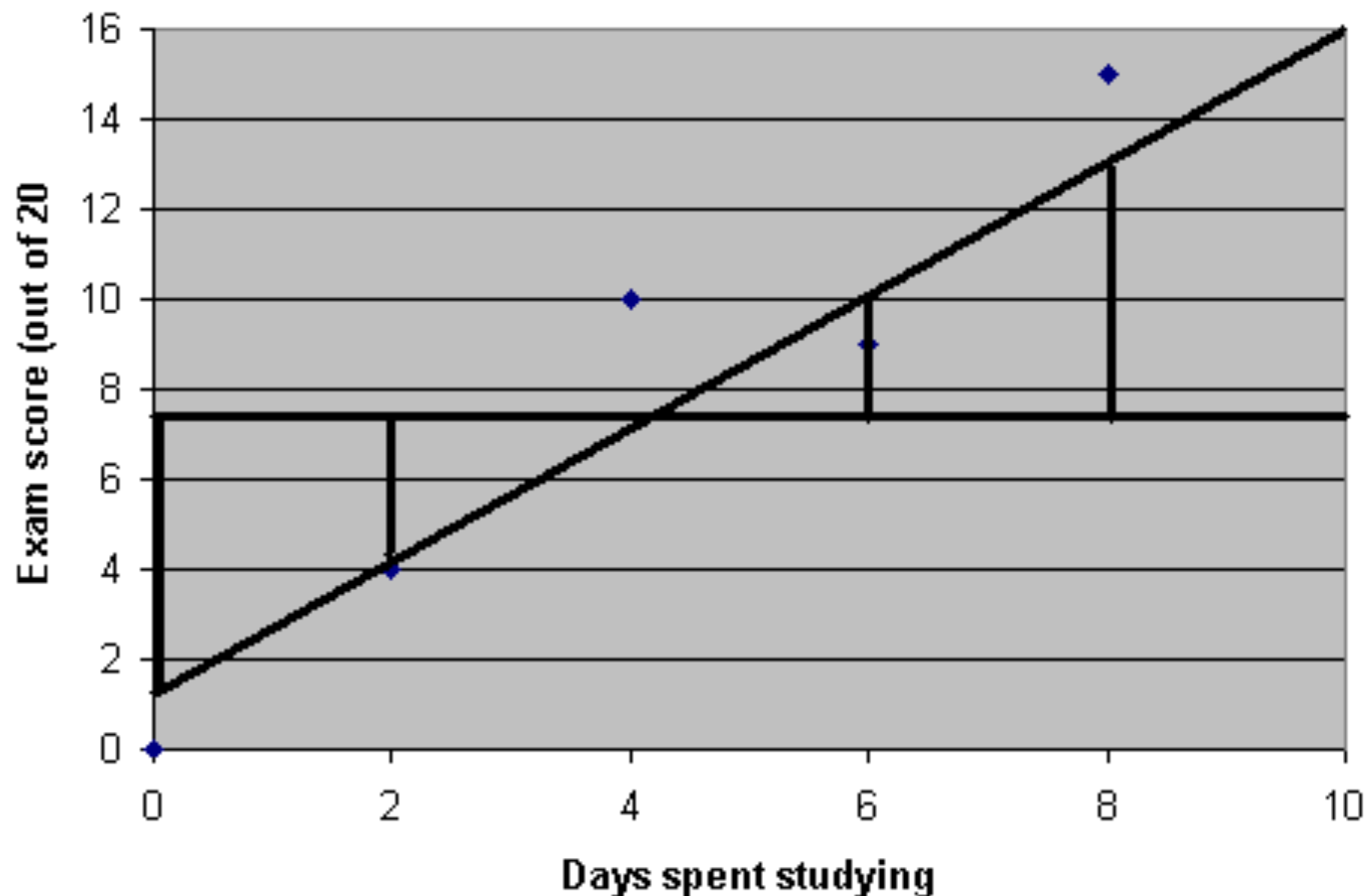
How much each observed value differs from the mean of  $y$  (called  $SS_T$ ):



How much each observed value differs from the regression line (called  $SS_R$ ):



How much the mean value of Y differs from the regression line for different values of x (called  $SS_M$ ):



- If  $SS_M$  is large, then the regression model is better than the mean in terms of predicting values of the outcome variable.
- If  $SS_M$  is small, then the regression model is not much better than the mean in terms of predicting values of the outcome variable.

- We can calculate the proportion of improvement in prediction by looking at the ratio of  $SS_M$  to  $SS_T$ .
- Actually, this is called  $R^2$  so:

$$R^2 = \frac{SS_M}{SS_T}$$

And this is the same  $R^2$  that we worked out by squaring the Pearson correlation coefficient.....

- We can also assess how good our model is by using the F-test.
- The F-test is based on the ratio of the improvement due to the model ( $SS_M$ ) and the difference between the model and the observed data ( $SS_R$ ).
- Rather than use the sums of squares themselves, we use the mean sums of squares ( $MS_M$  and  $MS_R$ ).

$$F = \frac{MS_M}{MS_R}$$

- A good model will have large  $MS_M$  and a small  $MS_R$
- In other words, the improvement of the model compared to the mean will be good.
- The difference between the model and our observed data will be small.

$$F = \frac{MS_M}{MS_R}$$

- If  $MS_M$  is large and  $MS_R$  is small, then  $F$  will be large.
- We can determine whether our  $F$  value is significant by looking up the critical values on the  $F$  table.
- For  $SS_M$  the degrees of freedom = number of variables in model (in our case 2).
- For  $SS_R$  the degrees of freedom = number of observations – number of parameters being estimated, including the constant (in our case  $5-2 = 3$ )



df numerator = 2, df denominator = 3 for our example.

df for numerator

df for  
denominator

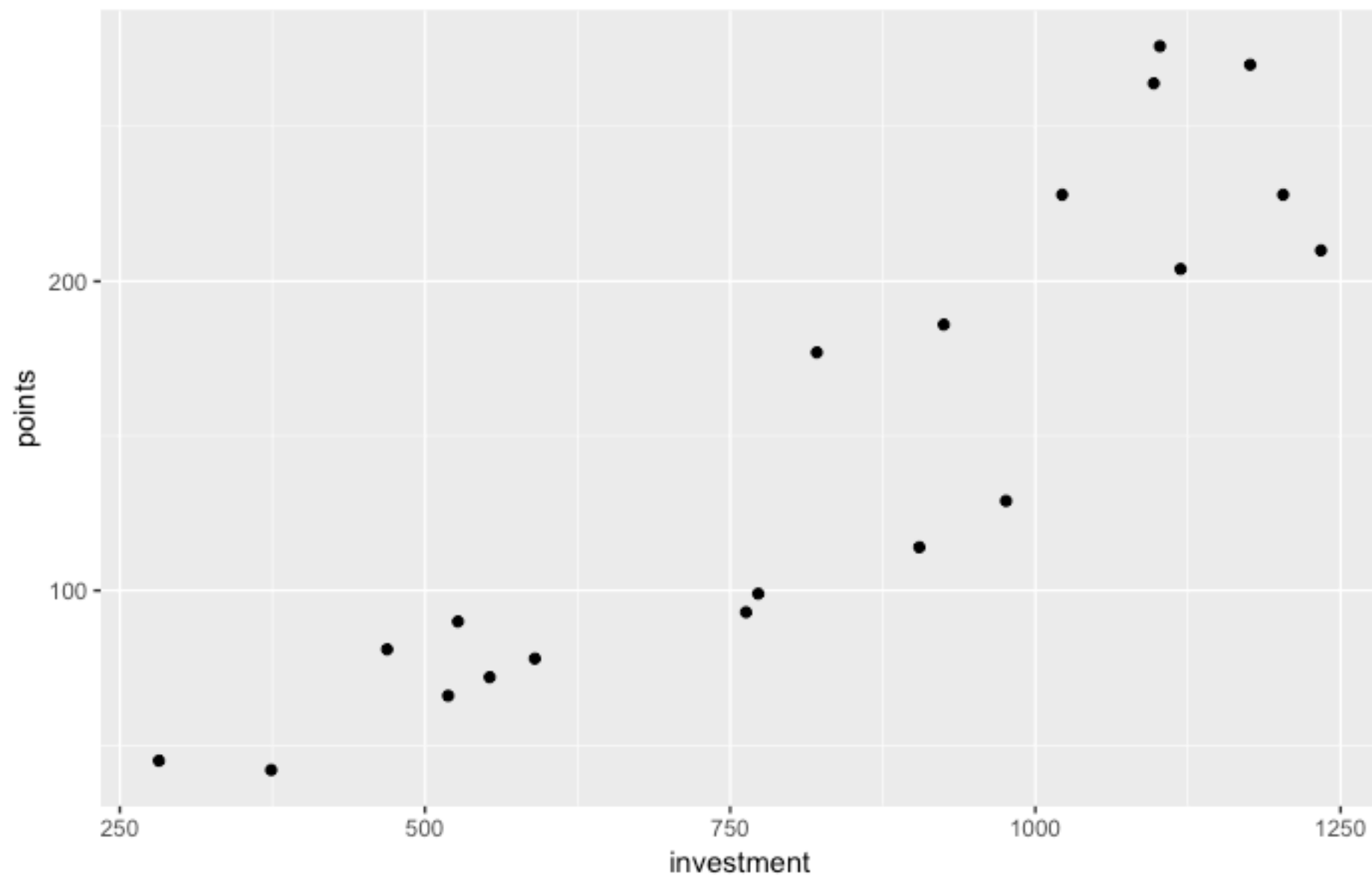
df2/df1	1	2	3	4	5
1	161.4476	199.5000	215.7073	224.5832	230.1619
2	18.5128	19.0000	19.1643	19.2468	19.2964
3	10.1280	9.5521	9.2766	9.1172	9.0135
4	7.7086	6.9443	6.5914	6.3882	6.2561
5	6.6079	5.7861	5.4095	5.1922	5.0503

So we would need an F value of greater than 9.5521 for our result to be significant at  $p < 0.05$

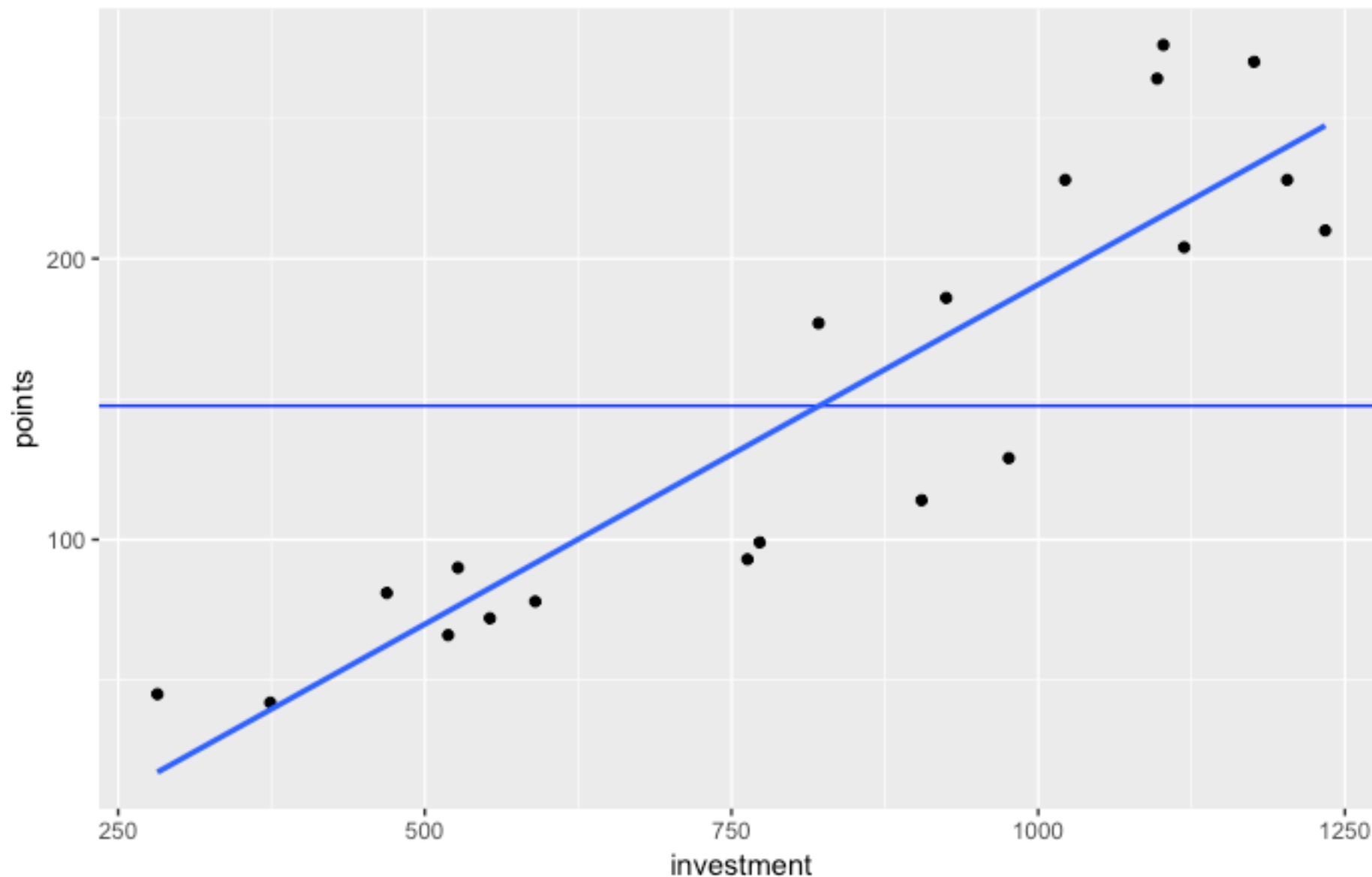
# An example

Imagine that you are Formula 1 team director. You're interested in understanding how the number of points that a team scores is predicted by the amount of money invested in the team. As well as being in charge of F1, you also have a secret interest in statistical analysis. In "dataset1" you will find (for each of the 20 drivers) the amount of money invested in their particular car (in £100,000s) plus the total number of points they were awarded over the season. Work out the simple linear regression equation that captures the relationship between investment (as our predictor) and points awarded (as our outcome).

```
> library(ggplot2) # For building ggplots  
> library(Hmisc) # Needed for correlation  
  
> #let's do a plot first  
> ggplot(dataset1, aes (x = investment, y = points)) + geom_point()
```



```
> # Let's add a regression line and a line of our outcome mean  
> ggplot(dataset1, aes(x = investment, y = points)) + geom_point() +  
  geom_hline(yintercept = mean(dataset1$points), colour = "blue") +  
  geom_smooth(method = "lm", se = FALSE)  
  
> # Let's calculate Pearson's r  
> rcorr(dataset1$investment, dataset1$points)
```



Pearson's  $r = 0.9$ ,  $p < .001$

# Building a simple linear model

```
> # Let's do regression with just the one predictor  
  
> model0 <- lm (points ~ 1, data = dataset1)  
> model1 <- lm (points ~ investment, data = dataset1)
```

We have built two models - *model0* is a model with just the intercept (so the mean of our outcome) predicting the outcome (*points*) while *model1* is a model with *investment* predicting the outcome (*points*).

```
> # You can compare the two models to each other  
  
> anova(model0, model1)
```

```
> anova (model0, model1)
Analysis of Variance Table

Model 1: points ~ 1
Model 2: points ~ investment
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1      19 120827
2      18  22046  1    98781 80.654 4.547e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The F-ratio comparing our two models is 80.654 indicating our model with our predictor (*investment*) is a better fit than our model with just the intercept (the mean).

```
> summary(model1)

Call:
lm(formula = points ~ investment, data = dataset1)

Residuals:
    Min       1Q   Median       3Q      Max
-55.936 -20.840  -2.978   28.212   60.615

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -50.92329    23.44967   -2.172   0.0435 *
investment    0.24166     0.02691    8.981 4.55e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 35 on 18 degrees of freedom
Multiple R-squared:  0.8175,    Adjusted R-squared:  0.8074
F-statistic: 80.65 on 1 and 18 DF,  p-value: 4.547e-08
```

Here we have our parameter estimates.

Here we have the t-test associated with our predictor (investment).

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

We would conclude from this that the amount of money spent on a driver does indeed predict the number of points they score in a season of F1. Specifically, for every £24,166 spent on them they will score one additional point.

Remember, regression is nothing more than prediction - a simple regression model allows us to predict the value of a variable future on the basis of knowing about that variable (and it's relationship to another variable) now...

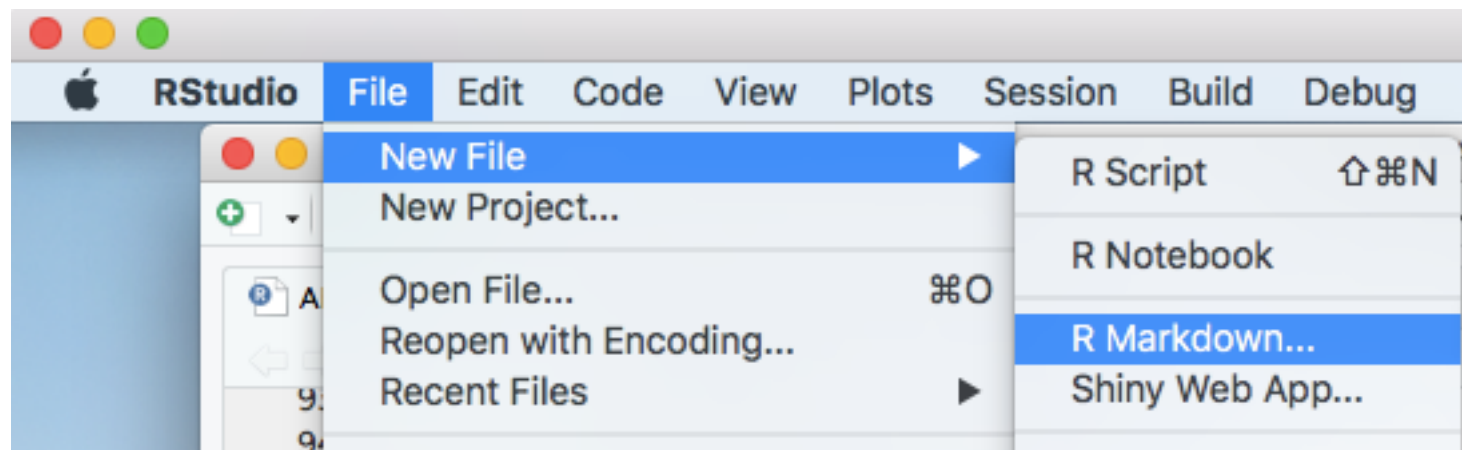
**To the worksheet\_1...**



# R Markdown

- R Markdown is a language in and of itself that allows you to produce documents in many formats (e.g., .html, .pdf, .doc) that contain your R code, the output of that code, and narrative that you write describing what you are doing (and why).
- R Markdown documents can be produced from within RStudio.
- R Markdown cheat sheet is in the R documentation folder.

- First we need to create a new R Markdown file:



Type the title of your document here.

New R Markdown

Document  
Presentation  
Shiny  
From Template

Title: ANOVA example

Author: Andrew Stewart

Default Output Format:

☒ HTML  
Recommended format for authoring (you can switch to PDF or Word output anytime).

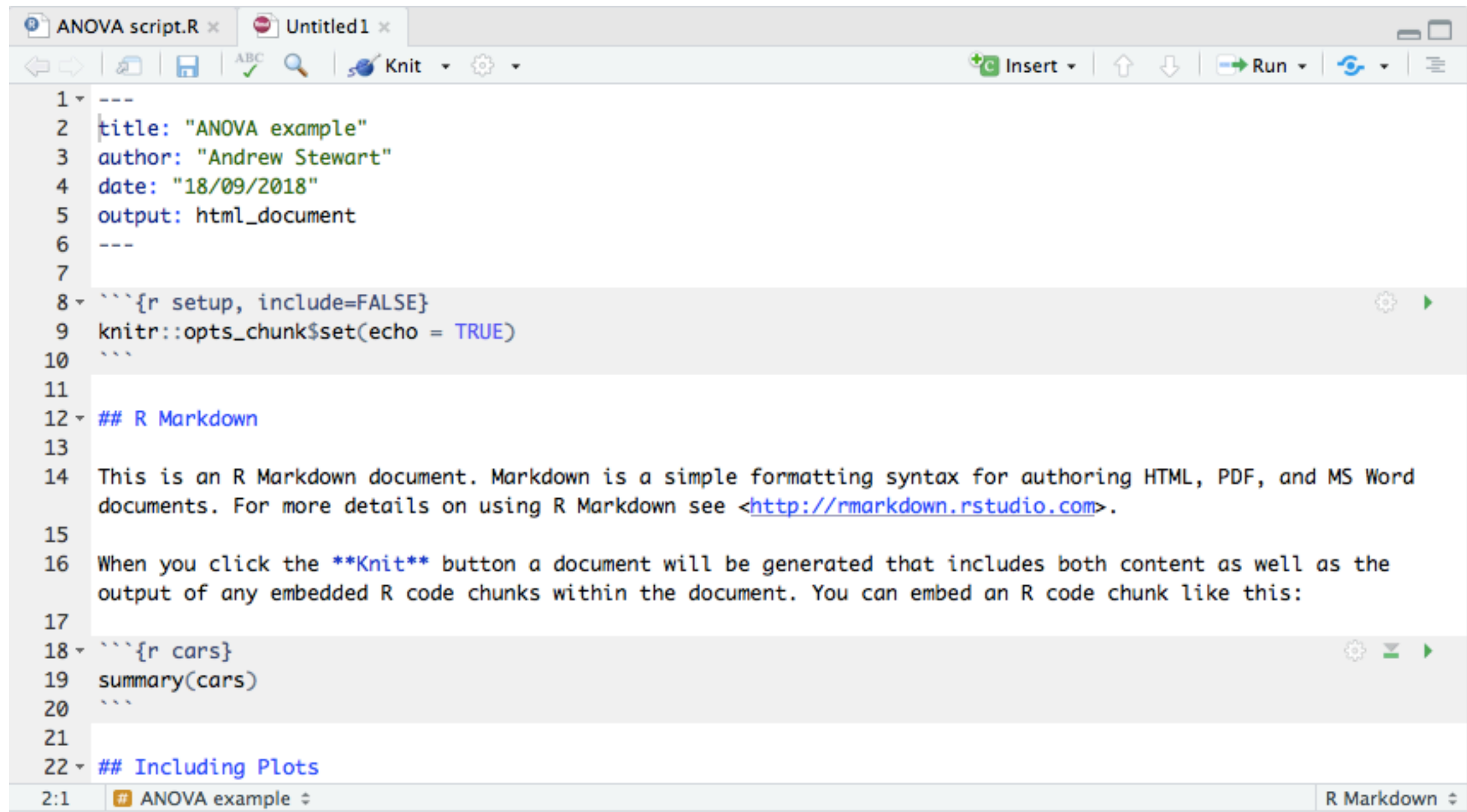
☐ PDF  
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

☐ Word  
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

OK Cancel


**Important** - replace your name here with your student ID number.

Select the kind of file you want to be generated by your Markdown - you can change this later btw.



```
1 ---
2 title: "ANOVA example"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word
15 documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the
18 output of any embedded R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
```

2:1 # ANOVA example R Markdown

You will now see a document like the above - it contains lots of example narrative (with a white background) and R code (with a grey background). We could actually 'knit' this document by clicking on  Knit to see what is produced...

What you will get is an html file (because that's the type of document we asked to be produced) that contains the R code, the associated output, plus the narrative. Notice how the ## symbols increasing the font size to allow us to generate headings in our narrative.

## ANOVA example

Andrew Stewart

18/09/2018

### R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

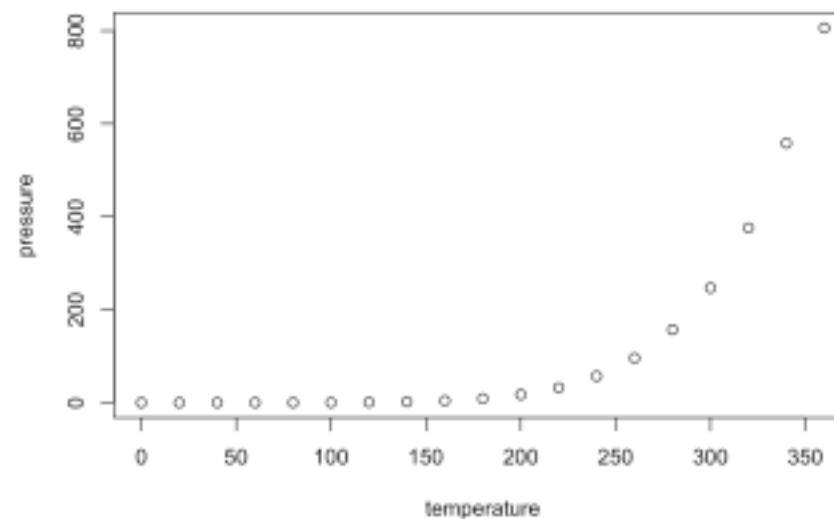
When you click the Knit button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
##  Min.   : 4.0    Min.   : 2.00
##  1st Qu.:12.0    1st Qu.: 26.00
##  Median :15.0    Median : 36.00
##  Mean   :15.4    Mean   : 42.98
##  3rd Qu.:19.0    3rd Qu.: 56.00
##  Max.   :25.0    Max.   :120.00
```

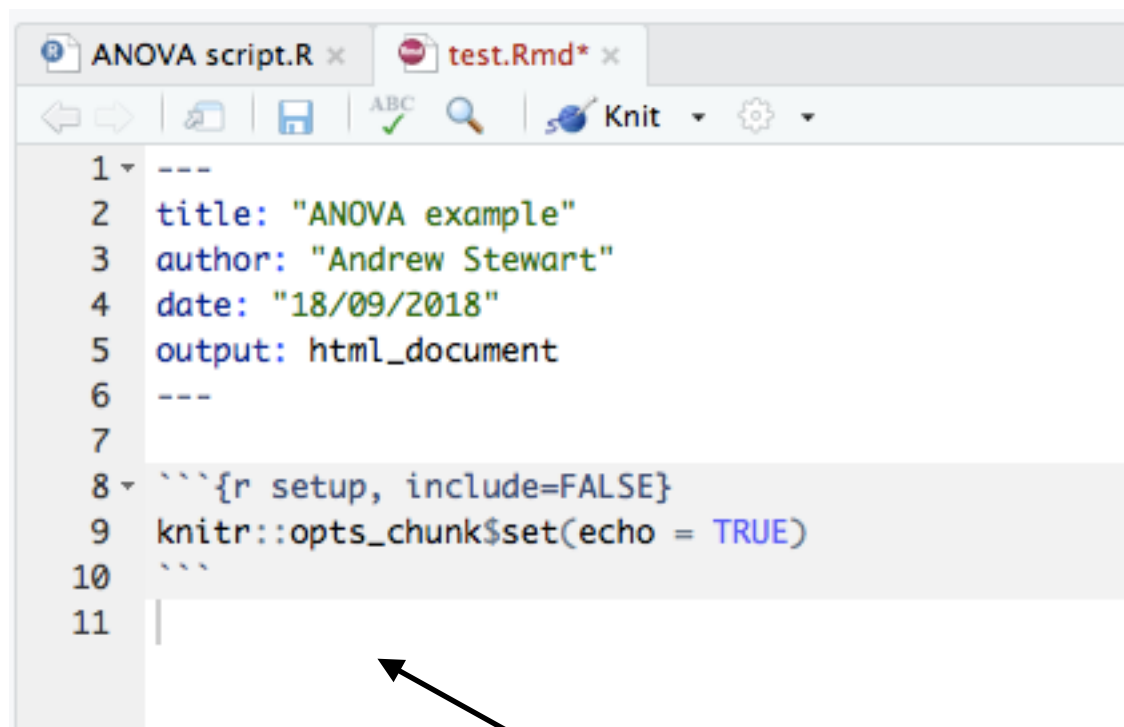
### Including Plots

You can also embed plots, for example:



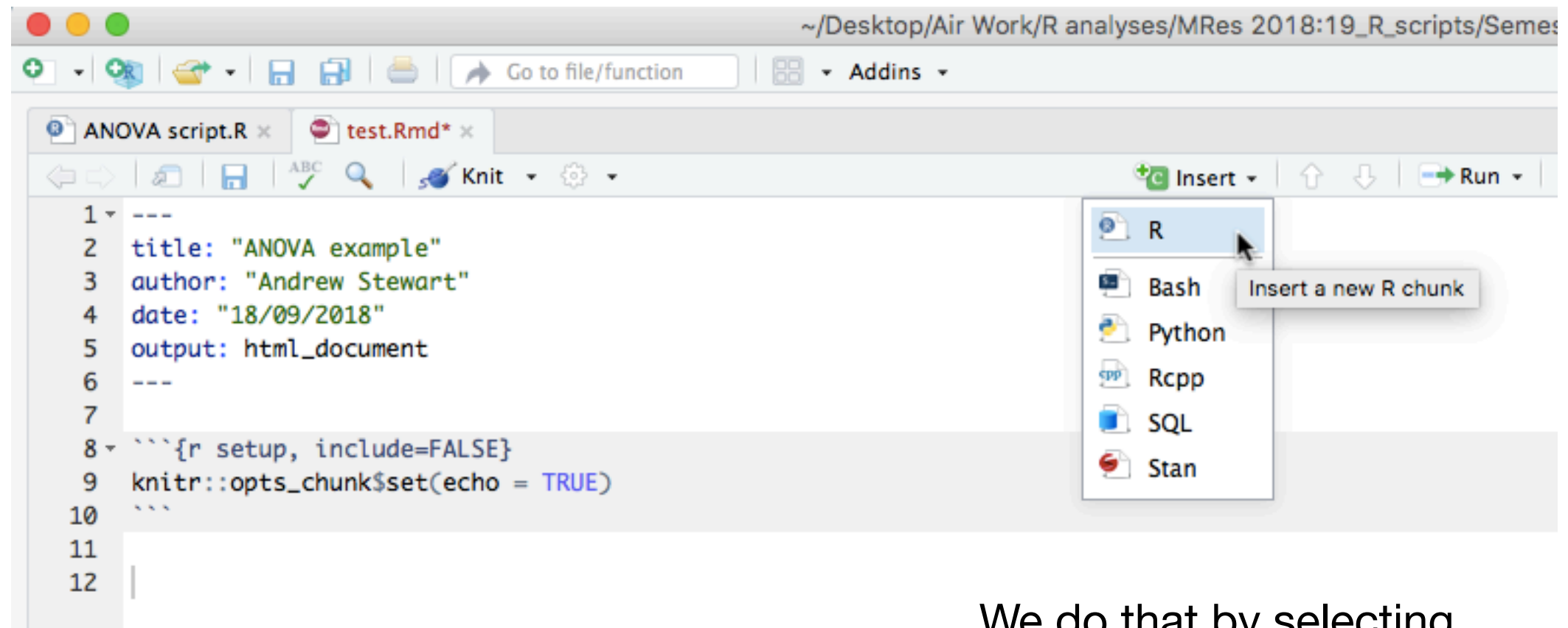
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

When you come to create a document using R Markdown, you'll first want to delete a lot of the example code that appears when you first start a new Markdown file:



```
1 ---
2 title: "ANOVA example"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
```

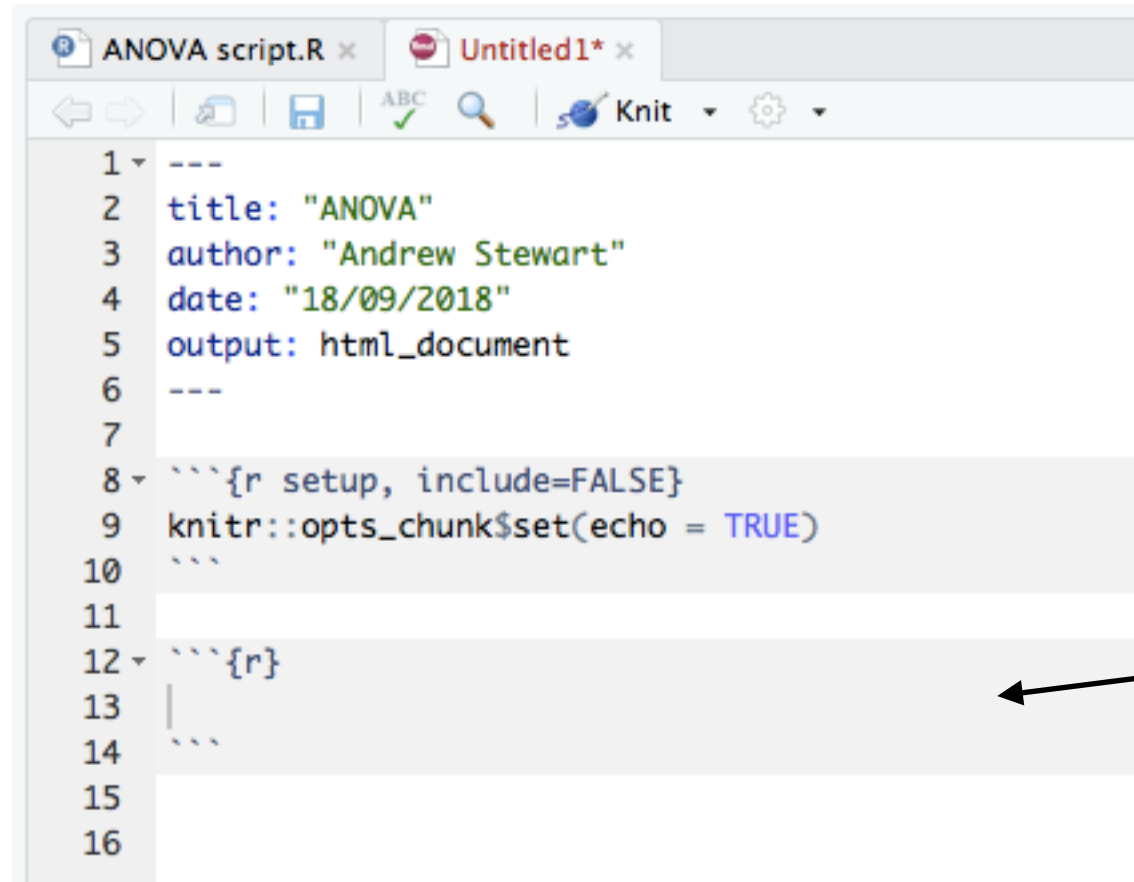
We now want to paste in chunks of our R code here



The screenshot shows the RStudio interface with a file named 'test.Rmd\*' open. The editor contains a YAML header and an R chunk setup. The 'Insert' menu is open, showing options for R, Bash, Python, Rcpp, SQL, and Stan. The 'R' option is highlighted, and a tooltip says 'Insert a new R chunk'.

```
1 ---
2 title: "ANOVA example"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 |
```

We do that by selecting “Insert” and then “R” - this will allow us to add some new R code.



The screenshot shows the RStudio interface with a file named 'Untitled1\*' open. The editor contains the same YAML header as the previous screenshot, but now has a new R chunk inserted at line 12. An arrow points to the new R chunk.

```
1 ---
2 title: "ANOVA"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ```{r}
13 |
14 ```
15
16
```

We can now insert our new R code here.

```
1 ---
2 title: "ANOVA"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ```{r}
13 library(tidyverse)
14 ```
15
16 |
```

We load the tidyverse packages first.

But maybe we should precede that with some narrative (and a heading) explaining what we're doing.

```
1 ---
2 title: "ANOVA"
3 author: "Andrew Stewart"
4 date: "18/09/2018"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## This is my first R Markdown document
13
14 First we load the tidyverse.
15 ```{r}
16 library(tidyverse)
17 ```
18
```



What happens if we now knit this short document?

# ANOVA

*Andrew Stewart*

*18/09/2018*

## This is my first R Markdown document

First we load the tidyverse.

```
library(tidyverse)
```


```
## -- Attaching packages ---- tidyverse 1.2.1 --  
--
```

```
## ✓ ggplot2 3.0.0      ✓ purrr 0.2.5  
## ✓ tibble 1.4.2       ✓ dplyr 0.7.6  
## ✓ tidyr 0.8.1        ✓ stringr 1.3.1  
## ✓ readr 1.1.1        ✓ forcats 0.3.0
```

```
## -- Conflicts ---- tidyverse_conflicts() --  
--  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()
```

Hmm, so we get some messages and warnings in our document - how can we get rid of these?

```
14 First we load the tidyverse.  
15 ```{r, message=FALSE}  
16 library(tidyverse)  
17 ```  
18
```



We can set the option for this chunk of R code not to display any messages by setting `messages=FALSE` within the first curly bracket. There are lots of other options available to use (e.g., `warnings=FALSE`).

Everything you need (and probably lots you don't) can be found in this cheatsheet here:

<http://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf>

and here is the 'R Markdown: The Definitive Guide':

<https://bookdown.org/yihui/rmarkdown/>

I've even put together a video tutorial talking you through how to make an R Markdown document...

**<https://youtu.be/CBJjxS-UopA>**

# Assessment

- The assessment needs to be produced using R Markdown.
- Top tips:

Have lots of narrative explaining what you're doing (and why).

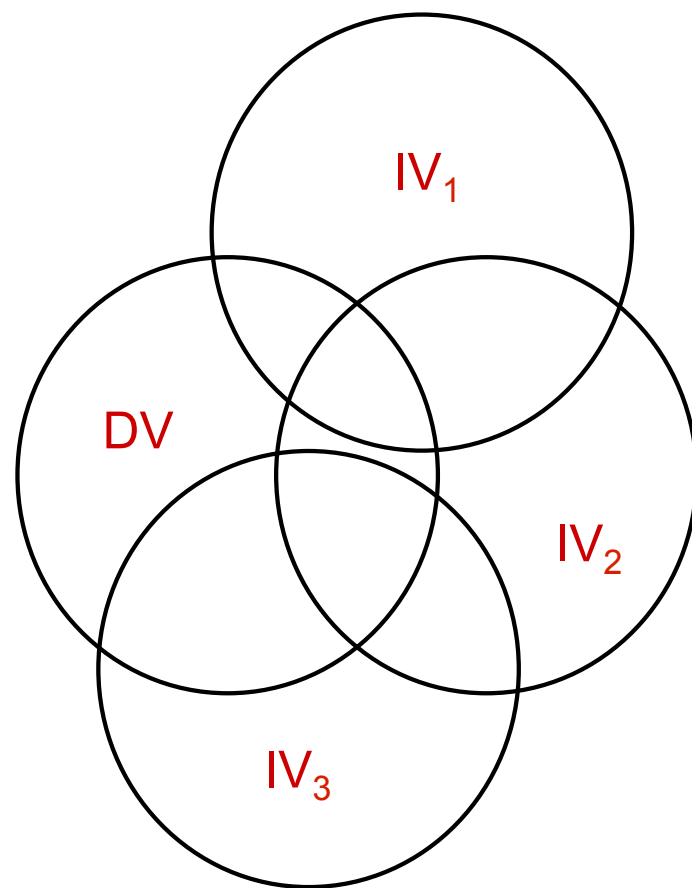
Use small chunks of code bookended by narrative - the ideal code chunk length is where you can have one comment that describes what that bit of code does.

If you find you're commenting about two things that one chunk of R code does, that chunk should probably be split into two...

# 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 residual error.

# Defining the Problem



To what extent do each of our IVs predict our DV (outcome)? Do we need to worry about our IVs being (too) correlated with each other?

# Data Considerations

- Sample size (do we have enough power?)
- Multicollinearity and Singularity.
- Errors of prediction are normally distributed: Skewness, Nonlinearity, Homoscedasticity.
- The influence of outliers and influential cases.

# How big should our N be?

- For testing the regression  $N \geq 50 + 8k$  (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: normal distribution of errors

- Errors of prediction are normally distributed around each every predicted ( $Y'$ ) score.
- That is, for every predicted score, the observed scores around that prediction should be normally distributed (i.e., normally distributed error).

# 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.
- Violation of this assumption is not dire – it weakens the power of the analysis to capture the relationship between the variables, but the analysis can proceed.

# 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.

# 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 one of our diagnostic plots).

# The Linear Model in R

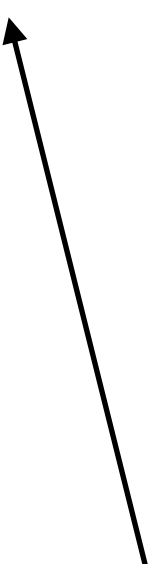
We are interested in whether house prices in 250 regions of the UK can be predicted by:

- (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)
Classes 'tbl_df', 'tbl' and 'data.frame': 250 obs. of 6 variables:
 $ Region      : num  1 2 3 4 5 6 7 8 9 10 ...
 $ House_price : num  193735 201836 191643 215952 203295 ...
 $ Population  : num  49004 48307 50379 53664 45481 ...
 $ Crime       : num  14 25 19 17 22 32 21 21 20 25 ...
 $ Average_age : num  72.7 78.1 71.4 72.1 76.1 ...
 $ Household_income: num  20843 19130 20411 16863 19964 ...
```

```
> head(my_data)
# A tibble: 6 x 6
  Region House_price Population Crime Average_age Household_income
  <dbl>   <dbl>         <dbl> <dbl>      <dbl>          <dbl>
1     1  193735      49004     14      72.7        20843.
2     2  201836      48307     25      78.1        19130.
3     3  191643      50379     19      71.4        20411.
4     4  215952      53664     17      72.1        16863.
5     5  203295      45481     22      76.1        19964.
6     6  191795      51582     32      81.2        20207.
```



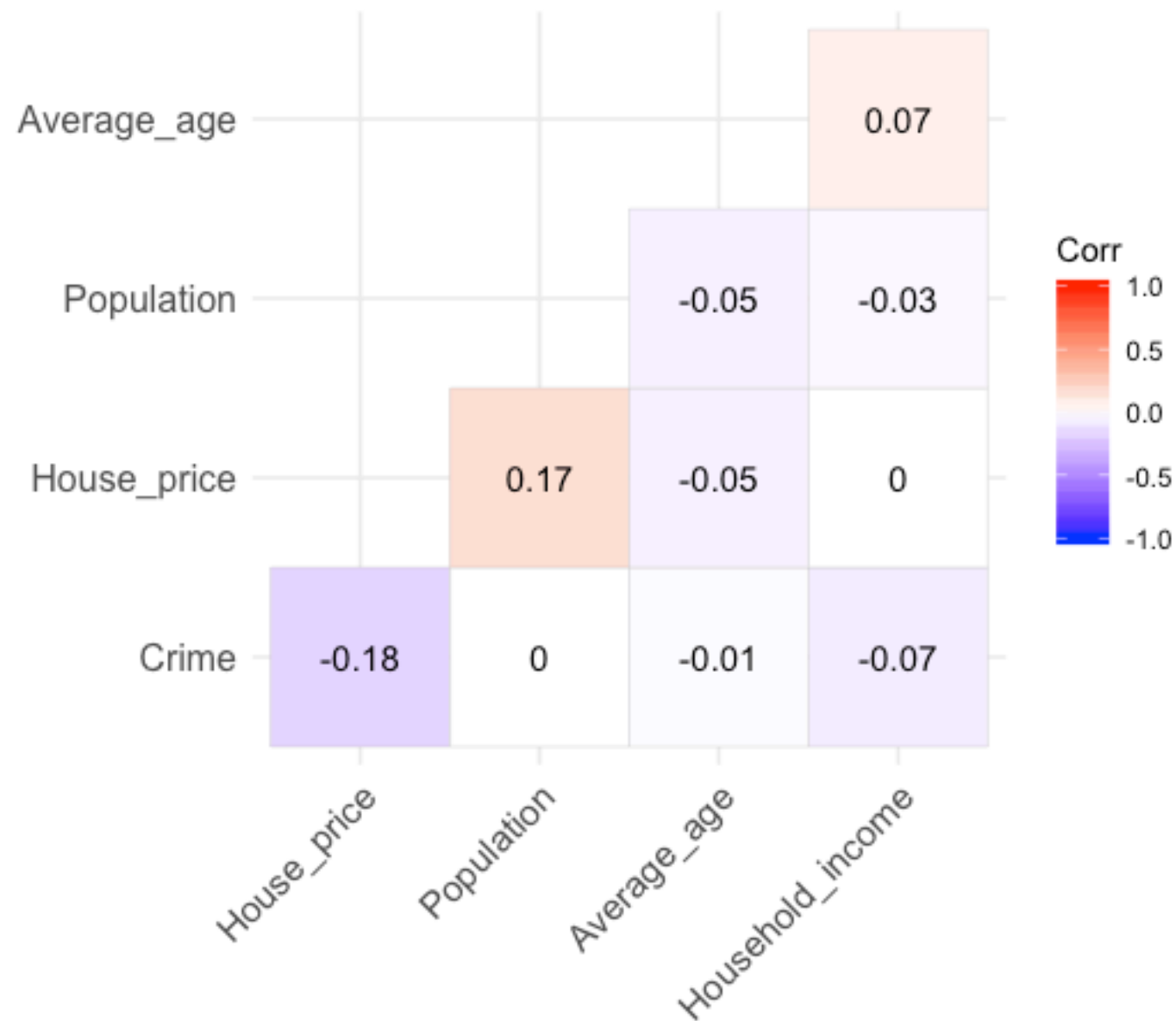
**Should change this to a factor**



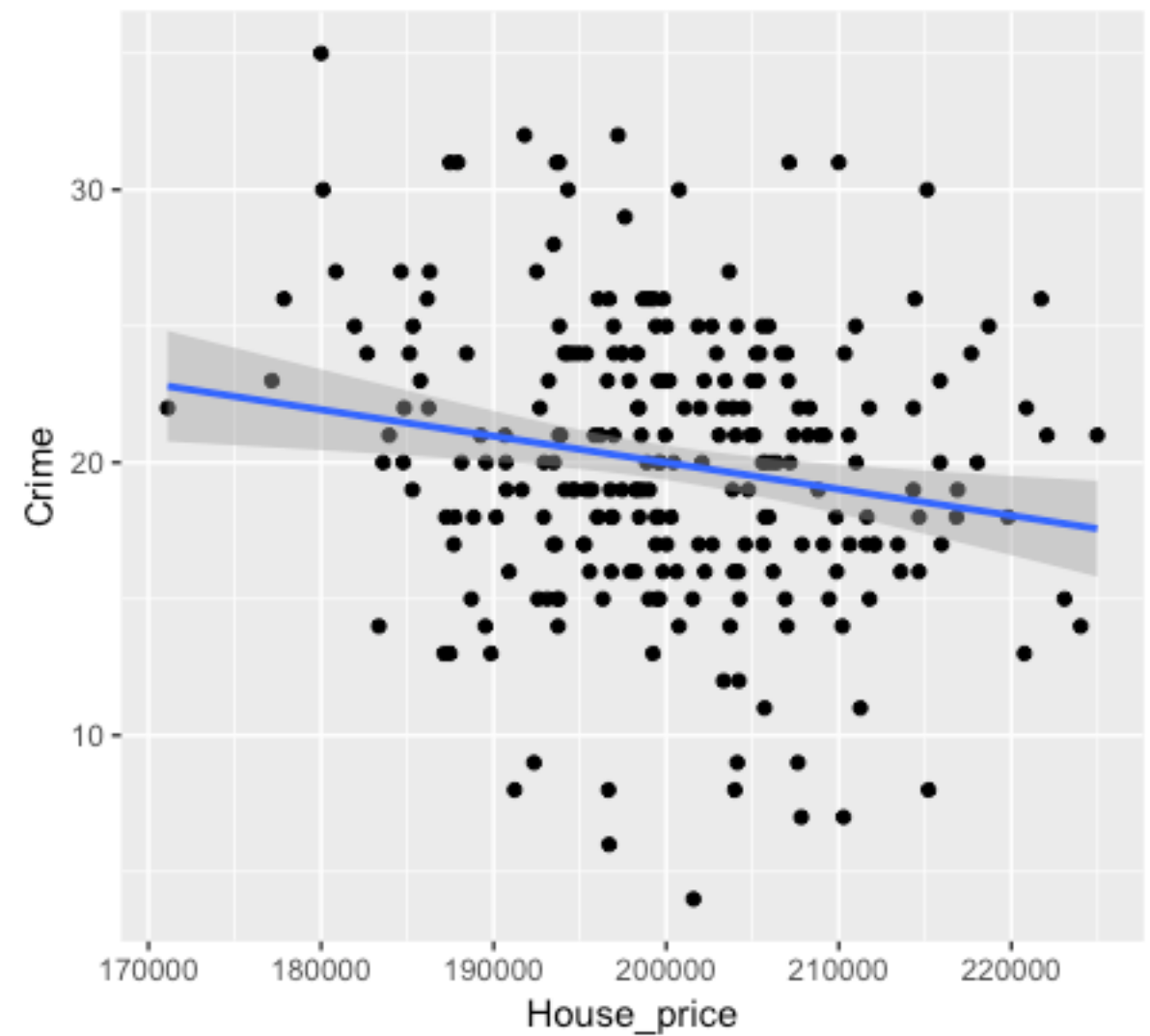
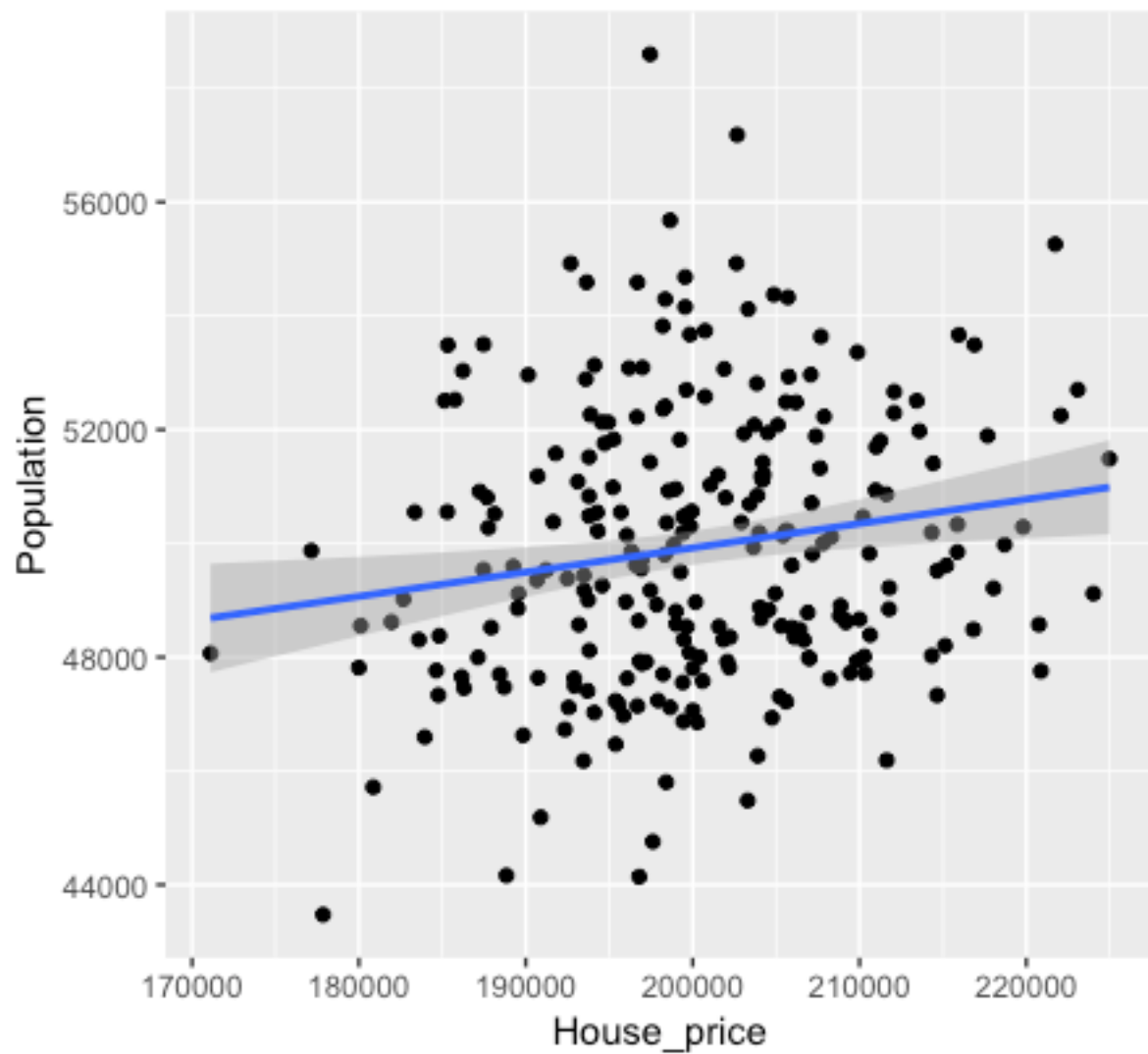
Let's build a correlation plot - need to install the package `ggcorrplot` first:

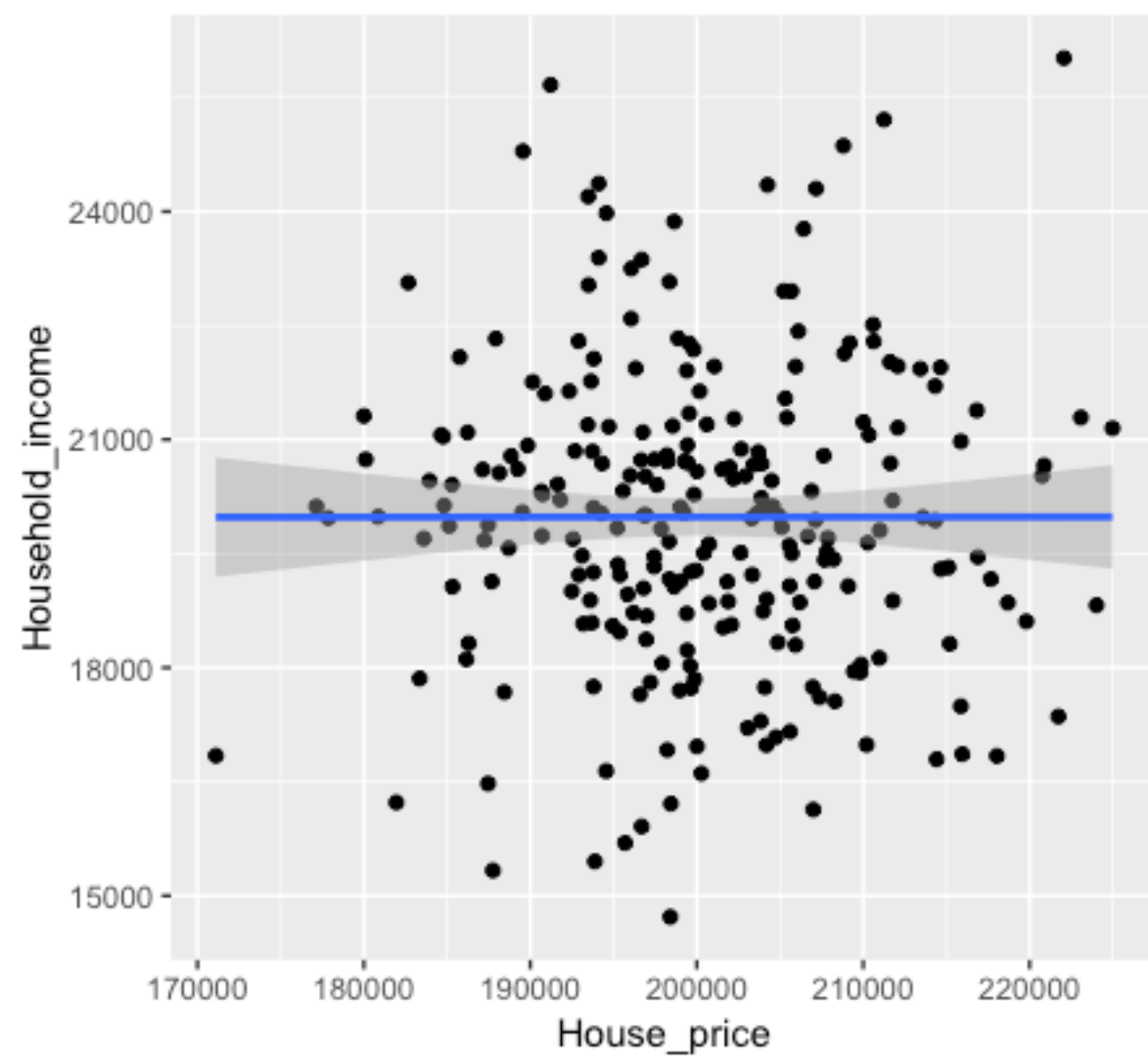
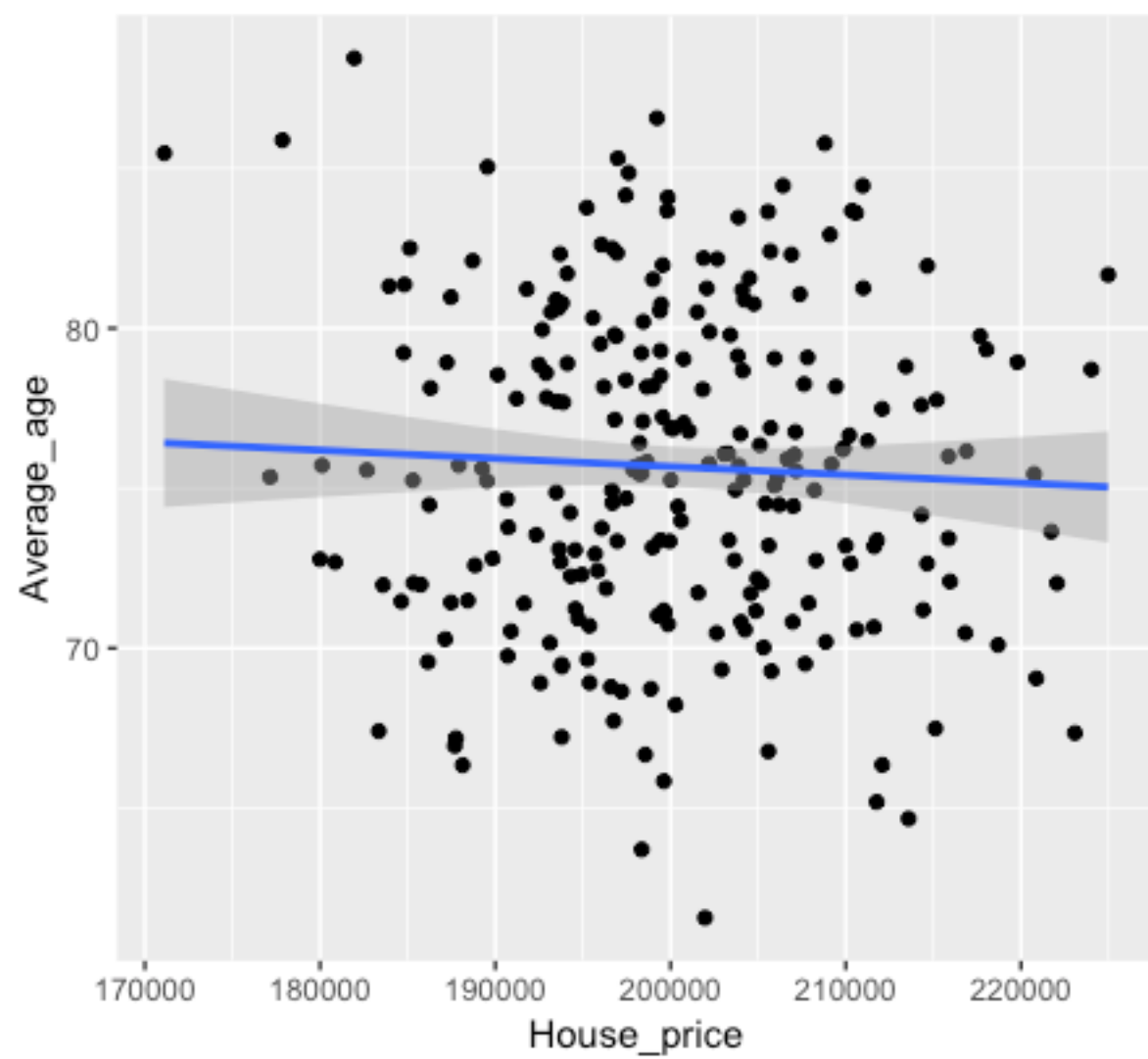
```
corr <- cor(dplyr::select(my_data, -Region))
```

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



Let's build some plots looking at the relationships between each predictor and our outcome variable.





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).

```
# 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

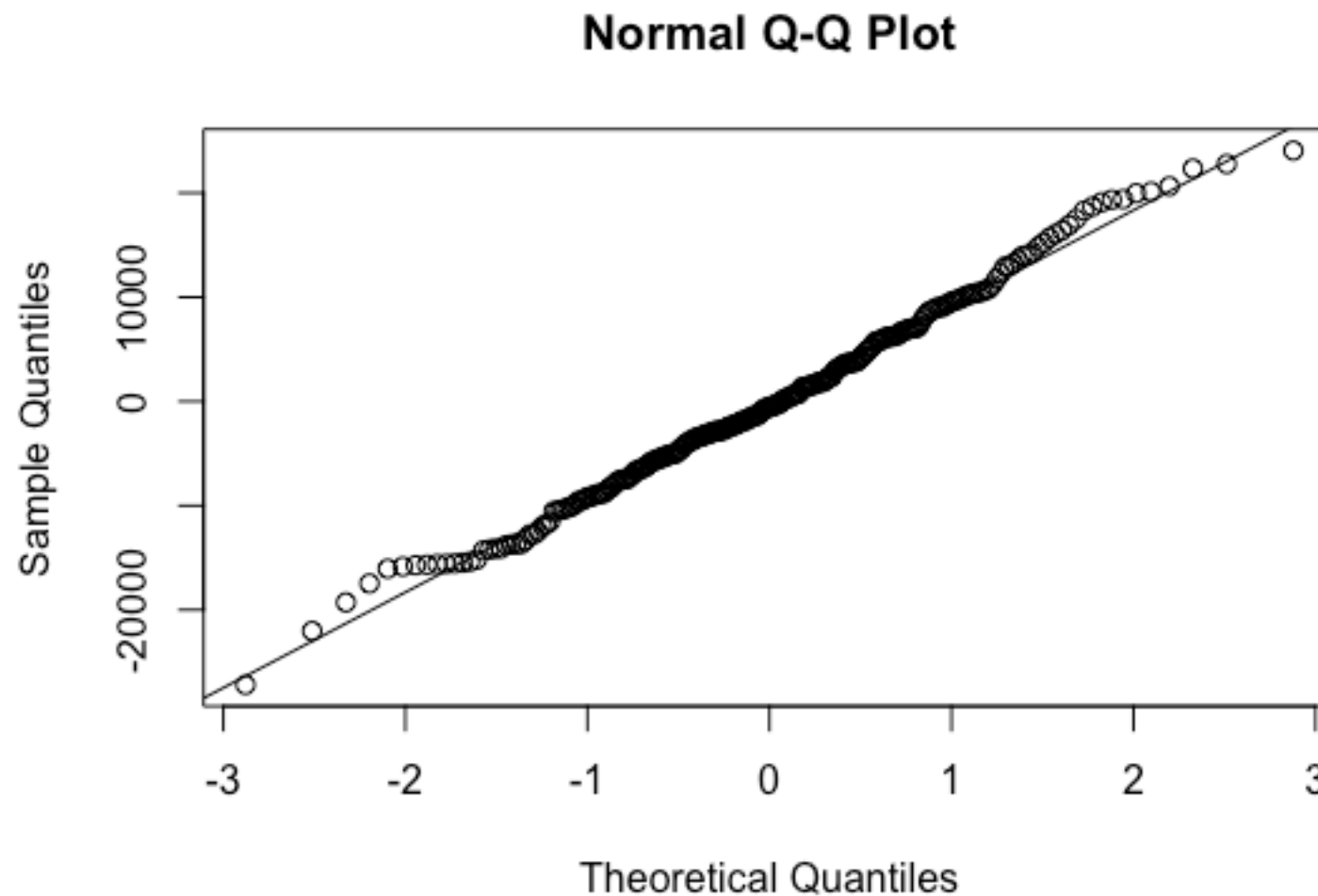
OK, so the models do not differ significantly by this test - we can use another measure of goodness-of-fit - AIC (Aikaike Information Criterion). AIC tells us how much information in our data is not captured by each model - lower values are better - can only be interpreted in a relative sense (i.e.,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

In the linear model, our residuals need to be normally distributed - the easiest way to check this is to plot them:

```
qqnorm(residuals(model2))  
qqline(residuals(model2))
```



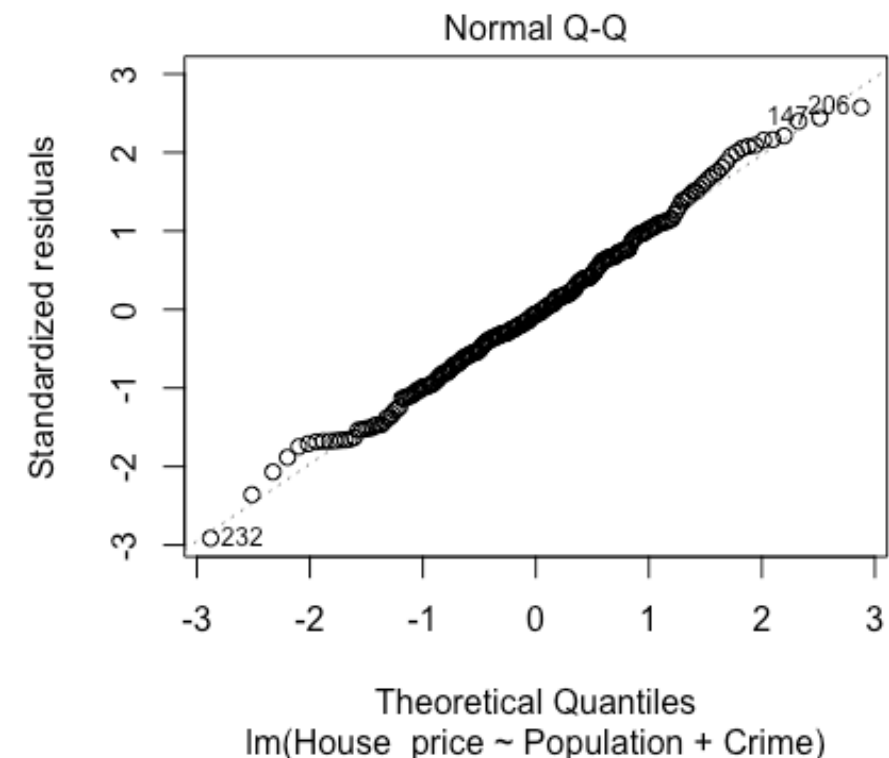
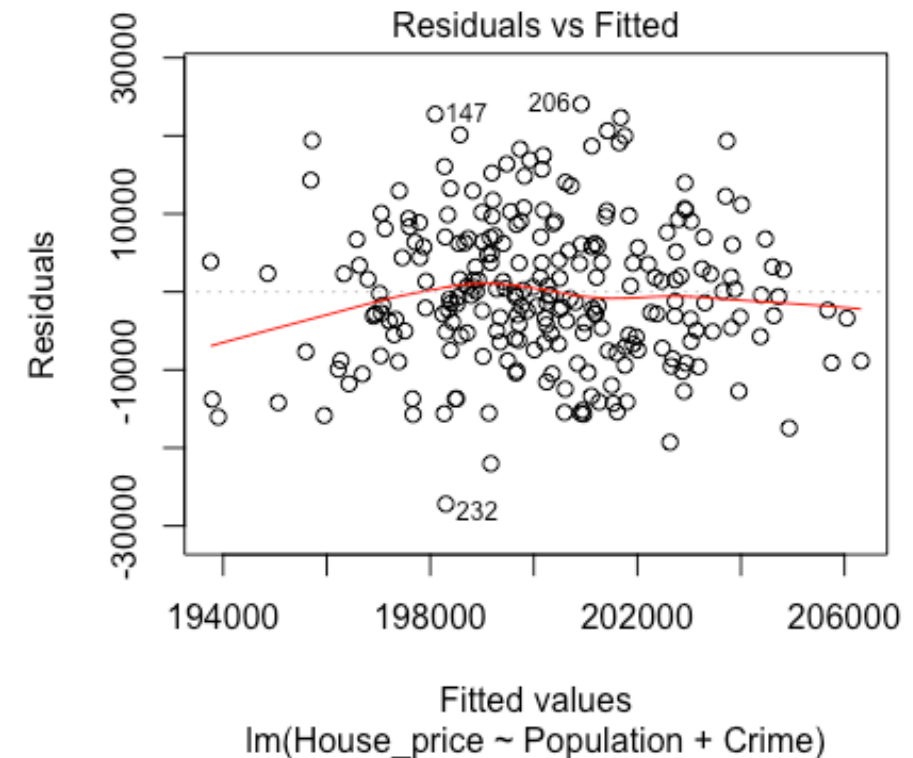
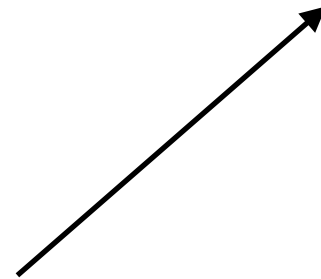
Now let's look at a number of diagnostic plots...



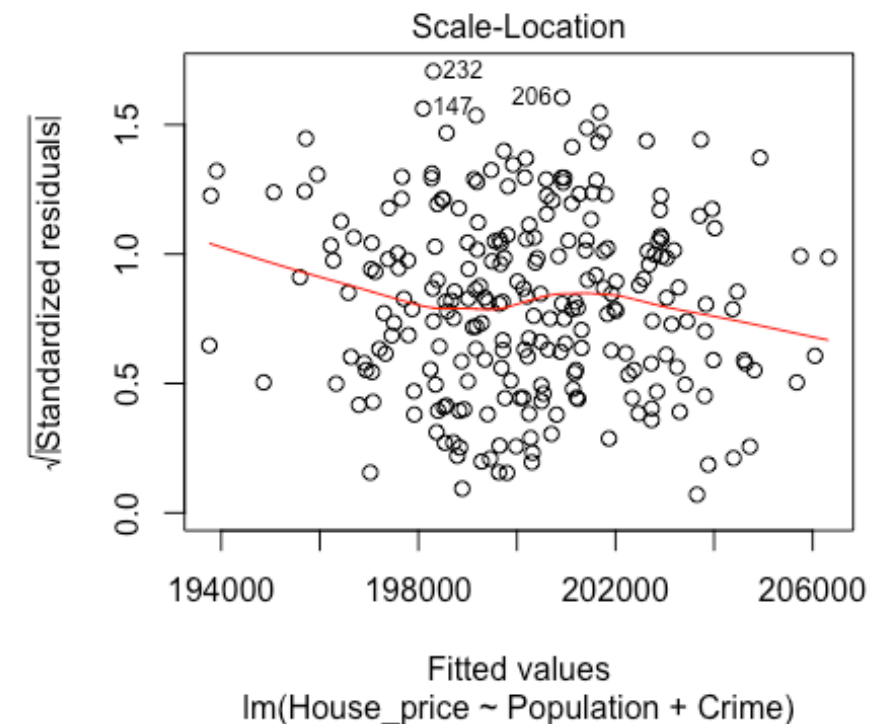
We can use the following command to get some visual representations of model fit:

```
> plot(model2)
```

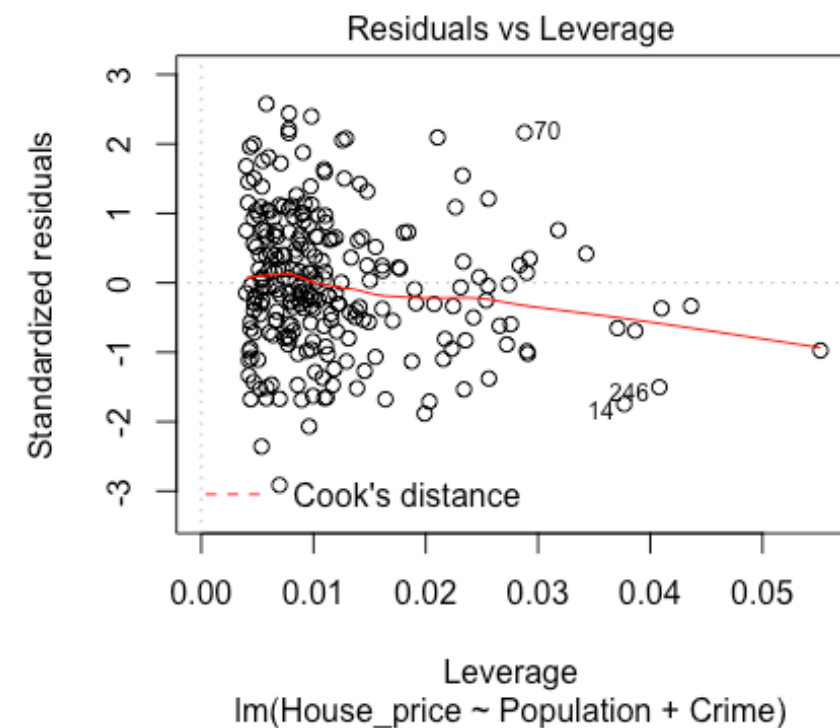
We should have a similar distribution of points (via LOESS Curve Fitting) either side of zero - if we don't it would suggest non-random errors (see Durbin Watson test later). In the Q-Q plot we should see a diagonal line if our residuals are normally distributed.



The Scale-Location plot shows if residuals are spread equally along the ranges of predictors. We used this to check the assumption of equal variance (homoscedasticity). We really want to see a horizontal line with equally, randomly spread points.



The Residuals vs. Leverage plot tells us about influential outliers (i.e., outliers that are affecting our model) - when cases are outside of Cook's distance (beyond the dashed line) it means they are having an influential affect on the regression model - we'd might want to exclude these points and rebuild our model.



# 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    0.66
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 -  $p = .66$  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")
```

- Let's focus on the combined method that adds predictors which improve model fit, and removes ones that don't - based on minimising AIC:

```
> summary(stepAICboth)

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(stepAICboth)
[1] 5286.855
```

We can see the procedure has settled on the model with Crime and Population. AIC value is 5286.855. In this case the stepwise model is the same as what we arrived at manually.

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(steplimitsboth, level = 0.95)
              2.5 %              97.5 %
(Intercept)  1.491596e+05 198110.856517
Crime        -5.602084e+02   -108.461481
Population   1.853052e-01    1.147126
```

# Stepwise Regression Based on Adjusted R Squared 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)
```

```
Forward Selection Method
```

```
-----
```

Candidate Terms:

1. Population
2. Crime
3. Average\_age
4. Household\_income

We are selecting variables based on p value...

Final Model Output

```
-----
```

Model Summary			
R	0.247	RMSE	9365.686
R-Squared	0.061	Coef. Var	4.678
Adj. R-Squared	0.053	MSE	87716066.079
Pred R-Squared	0.039	MAE	7416.676

```
-----
```

RMSE: Root Mean Square Error

MSE: Mean Square Error

MAE: Mean Absolute Error

```
-----
```

Parameter Estimates							
model	Beta	Std. Error	Std. Beta	t	Sig	lower	upper
(Intercept)	173635.236	12426.603		13.973	0.000	149159.616	198110.857
Crime	-334.335	114.679	-0.180	-2.915	0.004	-560.208	-108.461
Population	0.666	0.244	0.168	2.729	0.007	0.185	1.147

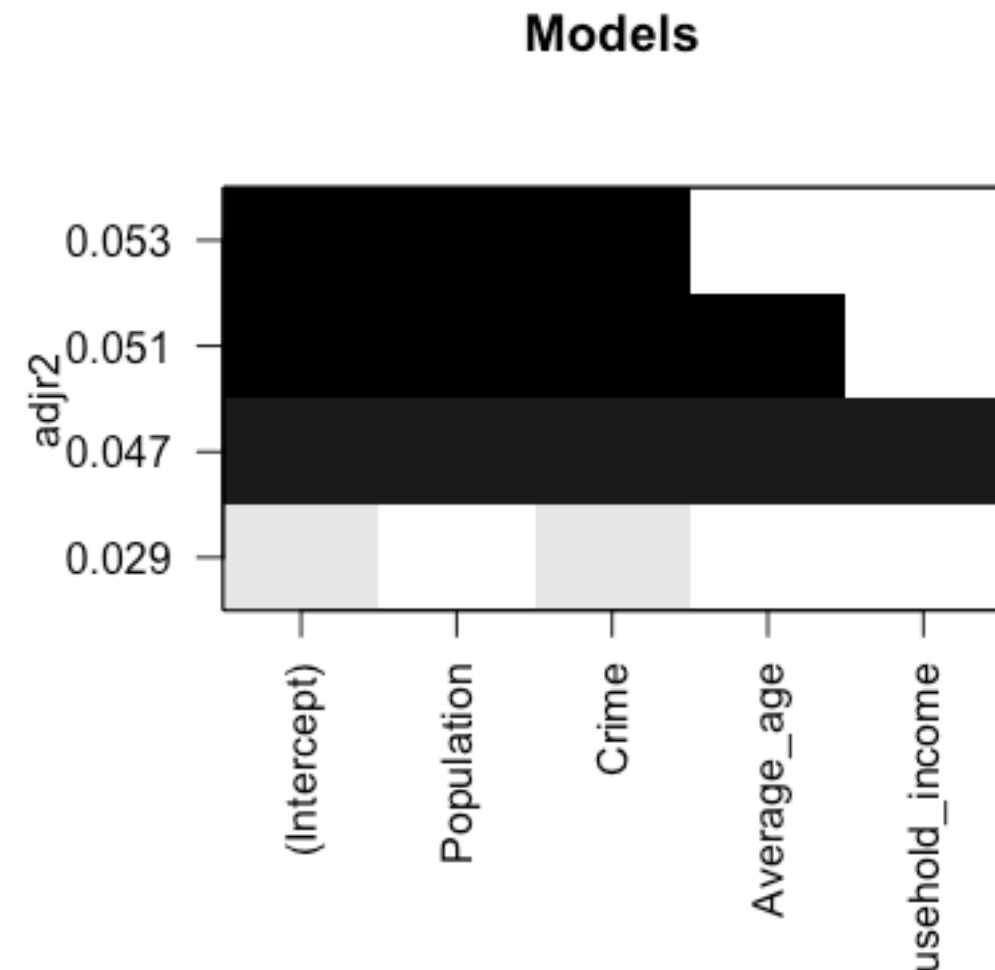
```
-----
```

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



- Visualise the possible models (incl. the one with the largest adjusted  $R^2$  value) using the *leaps* package.

```
> library(leaps)
> leapsmodels <- regsubsets (House_price ~
Population + Crime + Average_age +
Household_income, data = data)
> plot(leapsmodels, scale = "adjr2", main
= "Models")
```



# Collinearity?

- We can apply the `vif()` function to our model - it will work out the VIF values for each of our variables - `vif()` is in the `car` package so don't forget to load that...

```
> 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.

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 - 334.2 (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).

O'REILLY®



# R for Data Science

VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA

Hadley Wickham &  
Garrett Grolemund

You might want to consolidate your R knowledge by working through the R4DS book - freely available electronically at:

<http://r4ds.had.co.nz>

**To worksheet\_2...**