ST211 Course Notes
2021/22

Dr Sara Geneletti

Course details

Introduction

This section motivates the course, outlines the formative and assessed coursework as well as setting down my expectations around group work. I wont' cover everything in lectures so please read it in your own time.

Motivation

- Human beings generate and observe massive amounts of data.
  - Online Shopping/Loyalty cards
  - Visa/Passport/Citizenship Applications
  - Census/Longitudinal Studies
  - Political polls/Elections
  - University Applications
  - Medical records
  - Weather
  - Stock Market
  - Social Media
  - The list goes on and on...
- With advances in computer science we can store and analyse more data than ever before.
- We are in the era of *Data Science* ... which is just a fancy word for statistics (as is machine learning)
- One of the building blocks of statistics is *Regression Analysis*

Learning outcomes:

- *The aim of this course is to teach you how to perform linear and logistic regressions using* `R`*, one of the most widely used statistics programming languages*
- The course is hands on with a large programming component and very little theory (you cover that in ST300 if you want to)
- At the end of this course you will be able to:
  - Load and manipulate data in `R`
  - Perform, interpret and assess the fit of linear and logistic regressions in `R`
  - Have an idea of other types of regressions
  - Have an understanding of how and when to use transformations
  - Have an understanding of tools for model building, diagnostics and prediction

Expectations

I have high expectations of you!

- Coding can be difficult **especially** at the beginning. You will probably struggle and you will likely get frustrated.
- You are not alone and I expect you to interact with people in your group to understand errors and improve your coding
- Think of this experience as going to the gym. It's not supposed to feel good until you get better at it!
- I expect you to try and work out what the errors mean before you ask for help. By week 3 you should be able to navigate errors

- I expect you to finish all the Moodle quizzes in your own time if you do not do so in the time allotted in the workshop. If you find that you are unable to finish the workshops then think of them as valuable revision over the Easter holidays. I will reset all the quizzes at that point to allow you to re-do them
- I expect you to let me know at the beginning of workshops if you encountered any difficulties
- I expect you to attempt to work well with other people in your group and make sure any problems are brought to me as soon as possible
- *My expectations are no different from those of your future employers!!*

Asking questions

- *Always always always ask* if you don't understand something! We can't guide you through something if you don't ask!
- Chances are if you have a question others have the same one! You are doing everyone a favour
- Feel free to *interrupt* me during lectures/workshops to ask a question, I am always happy to answer!
- Phil, Antonio and myself use `R` regularly: think of us as coding coaches

Groups

- You will have received an email with details of your initial group – these may change a bit over the first two weeks of the term as students move in and out of courses
- You should be sitting with your group now (if not look for the table with your group number)
- You should stay in your group as these are people who you will be analysing data with for your projects – I will not change your group unless you have a very good reason.
- There is quite a lot of group work throughout the course and the two projects – reading week and end of year are with your group
- Make sure you exchange contact details (e.g. set up a whats-app group)
- At the end of this chapter is a list of rules of group work. **Please read them**

Details of the course

Format of weekly lectures/computer workshops

- 1 hour of lecture
  - The lecture notes are *gapped*
  - Watch the lectures BEFORE the workshop
- 2 hours of workshop in `R`
  - 1 hour with me
  - 1 hour doing a Moodle based quiz (or another activity) with Phil and Antonio to help out
- Course notes
  - You have a bound book of course notes: BE AWARE THERE MAY BE MISTAKES
  - The book has the weekly lectures and workshops as chapters
  - At the back of the book are also some excercises and special sections. I do except you to do them, either throughout the course or during revision as topics in these sections can come up in exams.
  - If there are errors/omissions I will issue corrections on Moodle

Coursework

There will be both summative and formative coursework. The reading week mini-project and the end of year project will all be based on the same data set *probably* from the Next steps longitudinal study.

Summative

- *Reading week group work project*: **10%** of the final grade. This assessment will:

  - Test basic understanding of how to perform multiple linear regression in ‘R including some diagnostics and interpretation
  - Serve as a test of the group dynamics
  - Introduce ‘R Markdown
  - Serve as a template for the end of year project
- *End of year group/individual work project*: handed out in week 7/8 and due in week 2 of ST
  - Group work worth 55%: Multiple linear regression (including 5% for meeting logs)
  - Individual work worth 35%: Logistic regression
  - This assessment will:
    * Test the ability to perform an in-depth regression analysis of a large dataset
    * Test the ability to work well in a group

Formative

- Weekly Moodle quizzes
- You should do these as they are relevant to the exam, the projects and contain examinable material that we may not cover directly in lectures

Moodle

- You'll find all the data sets and quizzes here.
- All the instructions for the projects.
- Instructions to get Zoom to work.
- A list of common error codes
- Weekly lecture capture
- You'll also find some videos to help you with difficult parts of the course and some materials that you will need to learn - e.g. how to derive the least squares estimates for simple linear regression.

R

- R is an open source (free, open to contributions from anyone, anyone can see how it works) statistics scripting/programming language
- A link is available on Moodle
- It is the most used language for statistics (although Python is becoming increasingly widespread)
- It has a large number of freely downloadable packages created by users to do almost anything in statistics
- It is relatively easy to use compared to e.g. C, C++ which are more complex programming languages.
- We will use RStudio, a versatile and user friendly interface for R

RStudio

- It is worth downloading RStudio onto your laptops whether you bring these to lectures or not
- The free desktop version is all you'll need.
- A link is available on Moodle

Laptops

- I encourage you to bring your laptop to the workshops.
- The lectures are designed to be followed without a laptop and while the R code will be included in the lecture notes from Week 2 onward I will not go over it in detail. That's what the workshops are for. I may make scripts available for some lectures – check on Moodle
- Whether you decide to bring your laptop to workshops or just use it for your projects you need to install R and RStudio on it.

- It is pretty straightforward to download everything, however if you have an issue feel free to come to my office hour or bring your laptop the workshop where myself of the GTAs can have a look.

#Rules of group work

Respect

Have respect for each other

- Respect each others ideas
- Respect the other group members
- Don't interrupt each other
- Everyone's opinion should count
- Be honest with each other especially if you don't understand something – explaining something can be as valuable for deeper understanding as having something explained to you.

Equal Participation/Contribution

All group members should do an equal amount of work

- Everyone should share the responsibility of the tasks
- Don't take over and don't let others take over
- Understand that different people have different ways of contributing to the work
- In both projects each person will write up what they contributed to and what % of the overall work they did as well as whether there was anyone who did not contribute sufficiently
- This will make it difficult for any individual to avoid doing their fair share of work.
- For the final year project I will also expect the group to hand in a plan of work to me by Week 10 detailing who will do what.
- If a student in a group is singled out as having contributed less I will ask them to come to my office and explain their contribution to the project to me and a colleague.

Common goals

Your group should have a common understanding of goals that need to be achieved

- Help each other to understand all concepts
- It is not acceptable to exclude a person from group work
- If you are feeling excluded or feel that others in your groups are being excluded please try and address this within the group
- If there is no change please let me know

Compromise/Co-operation

Be open to compromise

- Be willing to cooperate with others on their ideas
- Keep an open mind
- Vote on disagreements

Communication

Effective communication

- Make sure everyone is able to be vocal about their ideas and problems – this is especially important because not everyone is equally confident about expressing their ideas.
- If some people find it difficult to participate in discussions then they can write their ideas up in an email/chat
- Voice your ideas no matter how "off" you may think they are
- Listen effectively

- Don't be overly critical

Schedule

Time management

- Attend and arrive on time to all group meetings
- Be flexible about meeting times
- Keep on task (limit talk about non-related events)

Problems

Address problems quickly

- If there are issues e.g. unequal work-load, bullying, illness or other please come to me IMMEDIATELY!
- Depending on the complaint I will deal with this by contacting individuals in the group and/or the whole group.
- It is crucial that any issues are dealt with as soon as possible.
- **I will not tolerate any form of bullying or harrassment and will take action if any is reported to me.**

My details

- Dr Sara Geneletti
- Office hours: Monday 12:00-14:00. I take appointments on the student hub.
- Please make sure you book at least one hour in advance or I may not be able to see you.
- If you cannot come during my office hours because you have clashes then email me and we can find a suitable time.
- Office: 5th floor of Columbia House, room 5.07

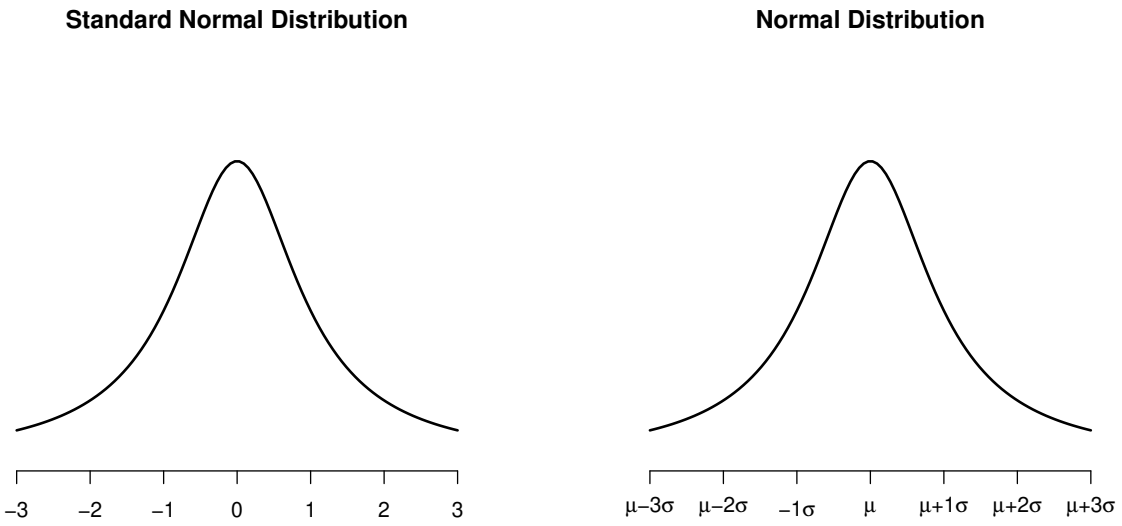
Lecture 1: Distributions, Histograms and Statistical Tests

Learning Outcomes:

Revision of the Normal and Student-T distribution, Hypothesis tests, P-values, statistical significance and confidence intervals

Normal distribution and Student-T distrubution

First off let's look at the Normal distribution



Parameters of the Normal distribution

Q: What are the parameters of the standard normal distribution? Left hand plot.

Q: What are the parameters of a generic normal distribution? Right hand plot.

Q: Explain what they mean.

The Normal (Gaussian) is typically used for tests and inference when the variance of a sample is known (but the mean isn't). However this is typically *not* the case

The Student-T distribution

The Student-T is used to model Normally distributed data when we do not know the mean or the variance. This is the most common situation. The Student-T has one parameter: the degrees of

freedom (typically the sample size-2). In most of the tests we conduct we have to estimate the variance from the data and so we can't formally use the Normal distribution but have to use the Student-T. In practice as we use `R` we don't actually need to worry about the details.

However for our purposes (understanding tests, p-values and confidence intervals) we can work with the Normal distribution as for a sample over 30 it is a close approximation to the Student-T.

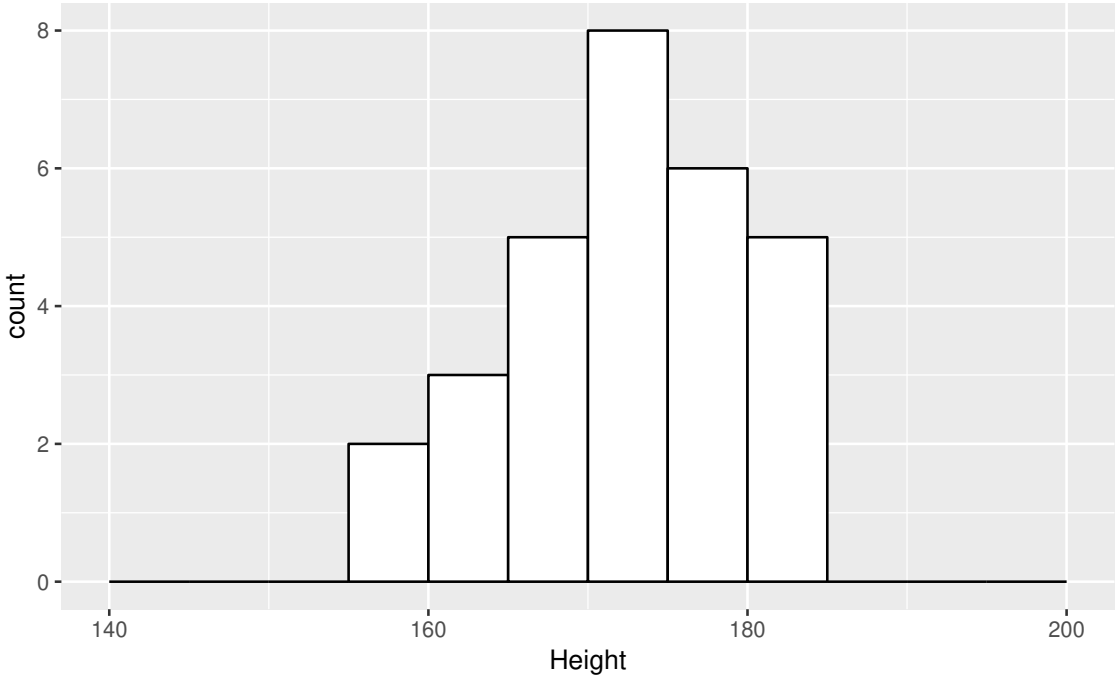
Histograms

Histograms are a way of looking at how the data behave for continuous variables. They divide the continuous variable into sequential sections (called bins) and display the number of times that the value of the variable falls inside each section.

29 students should have completed the Moodle questionnaire asking for their Height (in cm) and their Gender (Male,Female, Prefer not to say (PNTS).

The histograms for the heights is shown below:

- The x-axis of the histogram represents the range of possible values of height: 140 to 200.
- The bins marked in the plot below are in intervals of:
- The height of the bar in each bin represents the number of times the height falls into that bin: i.e. the frequency or count [how many students have heights between 175 and 180?:]
- The y-axis is therefore the number of people in each section – the count or frequency.
- Sometimes histograms show the probability (count/total) instead
- The histogram is a way to represent the *observed distribution* of the data



A summary of the heights is given below

```
print("summary")
```

```
## [1] "summary"
```

```
summary(df$Height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   157.0   168.0   175.0   173.2   178.0   185.0
```

```r
print("sd")
```

```
## [1] "sd"
```

```r
sd(df$Height)
```

```
## [1] 7.329339
```

```r
#plot(seq(140,190,by=5),seq(140,190,by=5),col="0",xlab="height (cm)", ylab="frequency",axes=FALSE)
#quantile(df2$height, p=c(0.025,0.975))
```

- The average height of students at the LSE is: 168 cm
- with *known* standard error: 7cm
- The average height of students in this class is:
- with *estimated* standard error:

Overlay a normal distribution with mean and standard deviation of LSE students.

Questions:

What questions can we ask about the heights of students?

First we want to know whether the height of students from the last 2 years of ST211 is the same (on average) as that of LSE students.

Q: Based on the statistics above, do you think the distribution of students at the LSE is a good match for the height of students in this class? Compare:

* The averages
* The spread

One sample z-tests

- We're going to make an unrealistic assumption for the sake of simplicity. The two variances above for all LSE students and those in this class are close enough so we'll assume that they are the same and that therefore the variance of height of students in this class is *known* to be:
- This means that we only need to look estimate *one* mean
- *If the averages are the same that means that the difference between them must be 0*

Hypothesis test

Formally the we talk about hypotheses for statistical tests. In this case

- The *null* hypothesis: There is no difference in the means $\mu_{class} - \mu_{LSE} = 0$
- The *alternative* hypothesis: There is a difference between them $\mu_{class} - \mu_{LSE} \neq 0$

Formula for the z-statistic

1. We subtract means in the numerator
2. We *standardise* in the denominator
3. This value is called the *z-statistic*
4. We compare it to the standard normal distribution

Intuitively if the absolute value of the z-statistic is very small i.e close to 0 then we can assume that the distribution of our observed heights is the same as that of the LSE student heights. If on the other hand the absolute value is large then it is far from 0 and it becomes harder to assume that the data from this class comes from the LSE student height distribution. The larger the absolute value of the z-statistic the less realistic the null hypothesis.

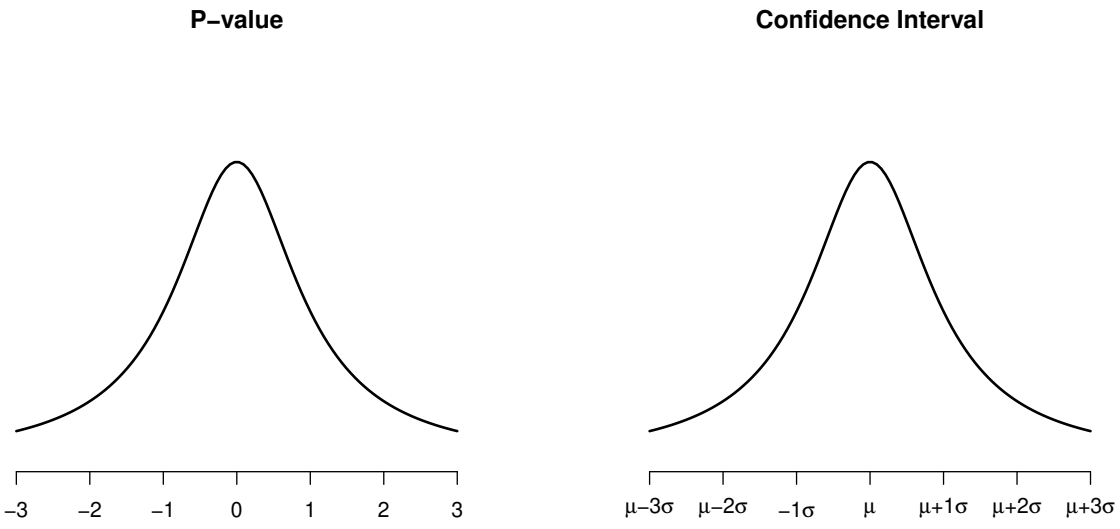
Somewhat arbitrarily, if the z-statistic exceeds 1.96 in absolute value we say that we have no evidence in favour of the null hypothesis at the 5% level. This corresponds to the 95% points. In practice we round this to 2 so we can make quick calculations.

Significance, P-value and confidence intervals

When the z-statistic is within the 95th quantiles then we say that there is *not sufficient evidence to reject the null hypothesis* and that the result is *not statistically significant at the 5% level*. If the z-statistic exceeds the 95th quantiles then we say there is evidence to reject the null hypothesis and the result is *statistically significant at the 5% level*.

The *p-value* associated with the test is the probability of observing the value we have or a larger value (in absolute terms): *provided the null hypothesis is true*. If the p-value is smaller than 5% (0.05) then we say that the result is *statistically significant at the 5% level*.

Another way of looking at it is to see whether the 95% *confidence interval* include 0. If it includes 0 then the result is *not statistically significant at the 5% level*. If the 95% confidence interval does not include 0 then the result is *statistically significant at the 5% level*.



Notes

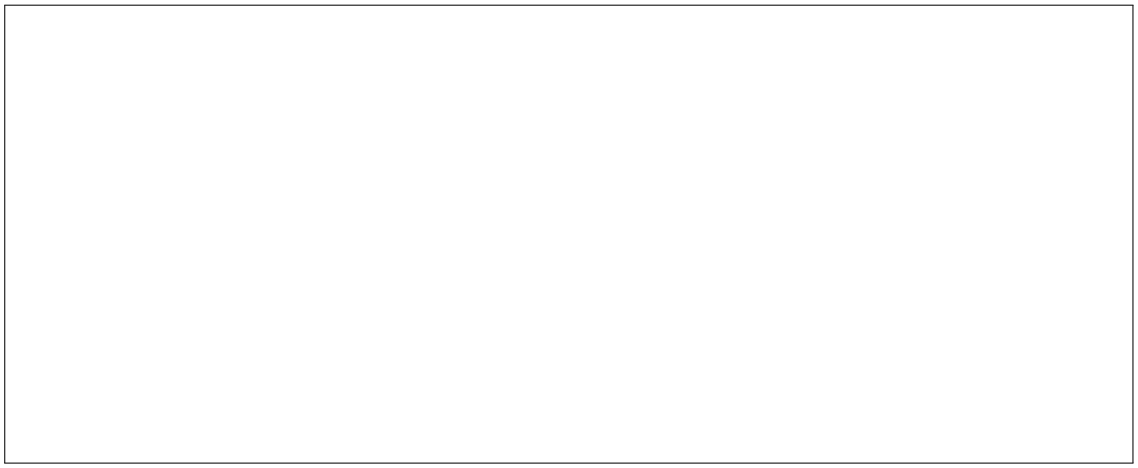
Typically we focus on the 5% level of statistical significance for 2-tailed tests. However we could consider different levels such as 1% or 10%. In some disciplines 10% significance is considered the standard. Also, we used a two-tailed tests which means that the 95% confidence interval is symmetric around the mean and there is 2.5% on either tail. However for some situations a one tailed test is appropriate. For example if we want to compare two values where the difference can only be 0 or positive it would make sense to do this (e.g. growth of children).

Two sample T-test:

The two sample t-test is similar to the one-sample test we saw above with some important differences:

1. We have to estimate the averages for both samples as they are both unknown
2. We have to estimate the variance from the data and therefore use a Student-T distribution
3. Typically we assume that the two groups have a common variance (this can be relaxed) and estimate a *pooled standard deviation*

Formula for the t-statistic and the pooled standard deviation:



We'll apply this to the data on heights where we look at men and women and ask whether their heights can be considered the same:

Males:

```
with(subset(df,Gender=="Male"), mean(Height))
```

```
## [1] 175.4545
```

```
with(subset(df,Gender=="Male"), sd(Height))
```

```
## [1] 5.396006
```

Females:

```
with(subset(df,Gender=="Female"), mean(Height))
```

```
## [1] 163.3333
```

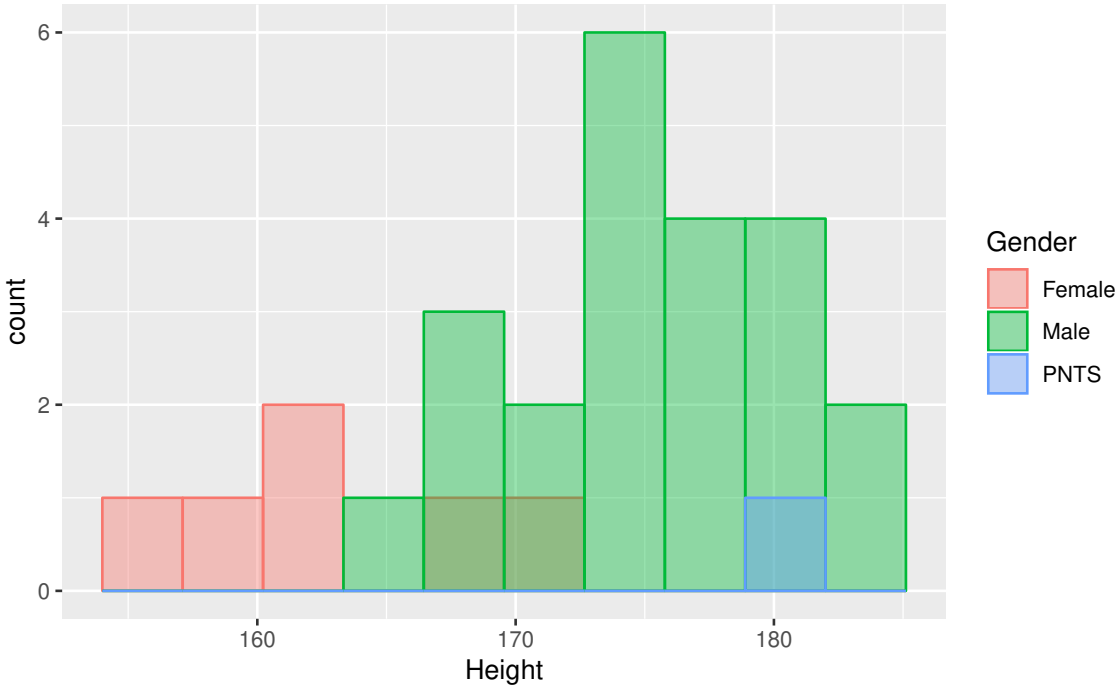
```
with(subset(df,Gender=="Female"), sd(Height))
```

```
## [1] 4.885352
```

- The average and sd of height of women in this class is:
- The average and sd of height of men in this class is:

Q: Calculate the (pooled) standard deviation bearing in mind that there are 6 Females, one Prefer not to say and the rest are males. There are 29 students altogether.

Below is the histogram showing men and women. Add two curves to the plot for the heights, one for men and one for women taking into account their means. The formula for the two-sample t-test with common variance is almost exactly the same as the one for the one-sample test and the principle is the same. If the means are the same then the distribution of the difference will be 0.



Q: Calculate the t-statistic bearing in mind that there are 6 Females, one Prefer not to say and the rest are males. There are 29 students altogether.

This value is called the t-statistic and is compared to the Student-T distribution on $n_w + n_m - 2$ degrees of freedom but we can interpret it in the same way as the z-statistic. Here n_w is the sample size of women and n_m is the sample size of men.

Hypothesis tests

Formally the we talk about hypothesis tests for statistical tests like the z and the t-tests:

- The *null* hypothesis: There is no difference in the means $\mu_m - \mu_w = 0$
- The *alternative* hypothesis: There is a difference between them $\mu_m - \mu_w \neq 0$

Q: Is the t-statistics we calculated significant at the 5% level? Why?

You should remember the formulae as you might be asked to calculate a t-statistic in an exam. In practice for our course we get `R` to do it.

Workshop 1: Introduction to R

Learning objectives

Learn about vectors and arrays in `R`

Familiarise yourself with `RStudio`

LSE PC alert The software and PCs are new and should work well. However please bear in mind that sometimes there are difficulties with the network and/or slightly different PC versions. In this case, please let me know asap although there is typically not a lot I can do immediately.

Why should I learn to code?

Good statisticians learn to program because:

- Independence: Otherwise, you rely on someone else having given you exactly the right tool (although we use many inbuilt *functions* or those developed by other programmers)
- Honesty: Otherwise, you end up distorting your problem to match the tools you have
- Clarity: Making your method is something a machine can do disciplines your thinking and means others can see what you're doing and reproduce it.

How?

We'll be learning how to do some programming and using built-in `R` functions.

- No programming knowledge presumed
- I expect you to know your 1st year stats (although we've been reviewing this).
- I drew heavily on a course by Cosma Shalizi and on Moodle I've put a link to his lectures on `R` if you want to know more.
- We will not necessarily cover all the material in this section but it is there if you want to use it for reference.

What now?

Today I will briefly go over the `RStudio` console.

- You will get a chance to practice more during the quiz
- If we don't get through everything I expect you to finish in your own time
- *It is going to be difficult* at the beginning. That is normal.
- You will make mistakes. It will be frustrating.
- It is important for you to read the *errors* you get – they are not always illuminating but you will come to understand them. There is a reference sheet to common errors and their meaning on Moodle.

`RStudio`

Go to the application → Statistics → `R` →`RStudio` and open the latest version with the "full libraries" option. Then go to File and New File → R script. `RStudio` allows you to see all parts you use for `R`:

- *The script*: where you write the `R` commands that work. You save this so you have a list of working instructions to refer back to – like a recipe or lego instructions
- *The console*: Basic interaction with `R` is by typing in the console, a.k.a. terminal or command-line. You type in commands, `R` runs them and sometimes gives back answers (or errors)

- *The Environment/History tabs* : if you define variables or upload data it shows in the Environment tab. History tab is a list of the commands so far and is useful if you want to figure out which command you used worked
- *The Files/Plots etc tabs* : Files in the local directory, Plots you might be creating, Packages that are installed, Help and viewer. Mostly you will focus on the Files and Plots.

The working directory

The directory you open `RStudio` from is the working directory. If you want to know where it is and what's in it. Go to the console.

What directory am I in?

```
getwd()
```

```
## [1] "/home/sara/Google Drive/Work/NewR"
```

What's in it?

```
dir()
```

At the beginning of every session:

- Open Zoom if you are using it.
- Create a folder/directory named Week x in your H drive. E.g. today you can create a folder called "Week 1"
- When we use data save all the data files from Moodle into that Week's folder
- Save your `R` scripts and plots into this file
- To get `RStudio` to work in this directory you use

`setwd(` `)` * Or in `RStudio` go to the `Files` tab in the bottom right hand corner and navigate to the directory you wand and then under `More` choose `Set As Working Directory`.

Data types

- *Booleans*: Direct binary values: TRUE or FALSE (0,1) in `R`
- *Integers*: whole numbers: positive, negative or zero
- *Characters*: single letters or symbols, strings = sequences of characters
- *Floating point numbers*: a fraction
- *Missing* or *ill-defined values*: `NA`, `NaN`, `Inf`.
- *numeric* includes both integers and floating point numbers

Operators

Type the examples with me:

- *Binary*: usual arithmetic operators, plus ones for modulo and integer division; take two numbers and give another number

Operator	Example	What it does
+	9+4	addition
-	9-4	subtraction
	9*4	multiplication
/	9/4	division
%/%	9%/%4	integer division
%%	9%%4	modulo

- *Logical operators* : Comparisons which are also binary operators

Operator	Example	What it does
>	9>4	greater than
<	9<4	less than
>=	9>=4	greater or equal
<=	9<=4	smaller or equal
==	9==4	equal to
!=	9!=4	not equal to

- *Boolean operators* : "and" and "or"

Operator	Example	What it does
&	(9>4) & (5<6)	are left *and* right true?
|	(9>4) | (5>6)	are left *or* right true?

- *Unary*: for arithmetic negation, ! `x!=y` would give true if x is not equal to y and false otherwise.

Checking/Imposing types

This can be useful when we don't know what the type of an object is or want to impose a type on an object.

- For asking use `is.`
- For imposing use `as.`

Example	What it does	Example	What it does (if possible)
`typeof(9.6)`	what type of variable?		
`is.numeric(9)`	is the variable a number?	`as.numeric(TRUE)`	makes the value numeric
`is.integer(6.7)`	is the variable an integer	`as.integer(6.7)`	turns value into its integer part
`is.character("a")`	is the variable a character/string	`as.character(6/5)`	makes a string of the number
`is.na(1)`	is the variable an NA		

See what errors you get when you do things that are not possible.

```
#as.integer("a")
```

Data objects

Data objects are the main thing we create, use and manipulate in `R`. We typically give data objects names. These are then called *variables*. We create them by using the assignment operator `<-` (or `=`). We do this because

- It makes it easier and faster to program
- Easier to debug
- Easier to build functions
- Easier to reuse and understand

```
new.var<-1
new.var
```

```
## [1] 1
```

```
exp(1)
```

```
## [1] 2.718282
```

It is also used to change the value of an existing variable

```
new.var<-5
new.var^2
```

```
## [1] 25
```

The Workspace/Environment

Look at the Environment tab on the top right of `RStudio`. Can you see new.var?

```
## [1] "new.var"
```

Should also just list `new.var`. If you change the view from list to grid you'll also see that you are given the type of the variable - `numeric()`.

Vectors

We can't do much statistics with a single value. Let's group some values together. The simplest *data structure* is the vector:

```
first.vector<-c(1,7,11,43)
first.vector
```

```
## [1]  1  7 11 43
```

```
is.vector(first.vector)
```

```
## [1] TRUE
```

The `c()` function creates a vector with the values in the order specified. Some things you can do to vectors

```
length(first.vector)
```

```
## [1] 4
```

```
first.vector+1
```

```
## [1]  2  8 12 44
```

```
first.vector*2
```

```
## [1]  2 14 22 86
```

```
first.vector^2
```

```
## [1]    1   49  121 1849
```

More things

```
second.vector<-c(1,2,1,2)
first.vector+second.vector
```

```
## [1]  2  9 12 45
```

```
first.vector^second.vector
```

```
## [1]    1   49   11 1849
```

```
first.vector>second.vector
```

```
## [1] FALSE  TRUE  TRUE  TRUE
```

To compare vectors use `identical` or `all.equal`

```
identical(first.vector,c(1,7,11,43))
```

```
## [1] TRUE
```

```
identical(c(0.3-0.1,0),c(0.4-0.2,0))
```

```
## [1] FALSE
```

```
all.equal(c(0.3-0.1,0),c(0.4-0.2,0))
```

```
## [1] TRUE
```

Extracting parts of vectors

If you want the 2nd value of your vector

```
first.vector[2]
```

```
## [1] 7
```

If you want the 2nd and 4th or the 2nd *to* the 4th

```
first.vector[c(2,4)]
```

```
## [1]  7 43
```

```
first.vector[2:4]
```

```
## [1]  7 11 43
```

If you want the vector *without* the 1st and 3rd

```
first.vector[c(-1,-3)]
```

```
## [1]  7 43
```

Say you want the values that are larger than 8

```
first.vector[first.vector>8]
```

```
## [1] 11 43
```

Say you want to know the position in the vector (the *index*) of the values that are larger than 8

```
which(first.vector>8)
```

```
## [1] 3 4
```

```
first.vector[which(first.vector>8)]
```

```
## [1] 11 43
```

Arrays

Let's make an array with 2 rows by 2 columns using our `first.vector`

```
first.array<-array(first.vector,dim=c(2,2))
first.array
```

```
##      [,1] [,2]
## [1,]    1   11
## [2,]    7   43
```

Can have any size array.

```
second.array<-array(first.vector,dim=c(4,2))
second.array
```

```
##      [,1] [,2]
## [1,]    1    1
## [2,]    7    7
## [3,]   11   11
## [4,]   43   43
```

How big is the array?

```
dim(first.array)
```

```
## [1] 2 2
```

Like with vectors we might want to know about individual elements of the array. See what the commands below do.

```
first.array[2,2]
```

```
## [1] 43
```

```
first.array[2,]
```

```
## [1]  7 43
```

```
first.array[2,,drop=FALSE]
```

```
##      [,1] [,2]
## [1,]    7   43
```

```
first.array[,2]
```

```
## [1] 11 43
```

Some commands will preserve the array structure

```
first.array>8
```

```
##       [,1] [,2]
## [1,] FALSE TRUE
## [2,] FALSE TRUE
```

Vector Recycling

If a row is longer than the vector that has been assigned to it, R will recycle the vector

```
another.array<-array(dim=c(4,2)) #another array
another.array[,1]<-c(1,2)
another.array[,2]<-c(3,4)
another.array
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
## [3,]    1    3
## [4,]    2    4
```

Functions

A *function* in R is a command that takes *arguments* and returns an *output*. A function works as follows: you type

`name.of.function(argument)`

and press enter then you will get the output of the function.

Example

```
which(first.array>8)
```

```
## [1] 3 4
```

Some functions will return vectors when applied to arrays if you don't set additional options. If you use `which()` with `arr.ind=TRUE` then it gives you the locations of the TRUE.

```
which(first.array>8,arr.ind=TRUE)
```

```
##      row col
## [1,]   1   2
## [2,]   2   2
```

Compare this to the output of `first.array>8`

The homework this week is to work through some code that teaches you about loops and functions in R

History

Have a look at your History tab. It contains ALL the commands you've typed so far (including the wrong ones). If you want to save your commands then it is easy to just save the History. You can remove individual commands by highlighting and clicking on the page with the red cross at the top of the tab.

Packages

Packages typically contain a number of (often related) functions to do things in `R`. One of the great things about `R` is that you can write your own functions but also you can often find a function someone else has written to do almost anything you need to do. Because `R` is open source you can access the code of any function anyone has written and modify it if you want. There are hundreds of packages. Go to the `R` website to search for packages or search via the packages tab in `RStudio`.

When we first open `RStudio` it has only a few packages loaded. Look under the the `Packages` tab in the bottom right hand corner of `RStudio` to see what's packages are installed. You load a package by typing `library(nameofpackage)` or by ticking the box next to the name of the package you want in the packages tab.

An important point is that `R` on the LSE PCs has all the relevant packages installed when you log in as you because it has been set up specially. So you should be able to load the packages without having to install them. Don't try to install anything as this can sometimes lead to crashes. If you install `R` on your laptop you'll have to install everything yourself. Use the install button in the `Packages` tab. If you encounter problems with some of the packages let me know asap and I can come up with alternatives.

At the end of every Workshop

- Save the `R` script in the working directory
- Save any plots in the working directory (use the "Export" tab)
- When you close `RStudio` it will prompt you asking if you want to save the `.Data` file. This is the "image" which contains all the variables, data and functions you have created during the session. It does *not* contain the script i.e. the set of instructions you typed, only the stuff you created.
- It is worth saving the image if
  - You will continue to work on the same project later
  - If the script takes a while to run and it is easier just to save the complex/large objects you have created
  - If you have manipulated the data in complex ways (in this case also save the data in a new data file)
  - If you intend to share your script with others it is important that it will run with an empty session – passing them the image will help with that.

Lecture 2: Variable roles and types, Useful plots

Learning outcomes

To become familiar with different types variables involved in regression

Reminder of boxplots and scatterplots

Introduction to Q-Q plots

The aims of regression

The principal aims of regression are to use observed data and maths to create models to:

1. Better *understand* and *explain* the relationships between different factors or variables
2. To *predict* the values of a variable of interest based on information currently available
3. A combination of the above

In order to do this we need to be able to classify data into *roles* and *types*.

Roles and types of variables

In regression we have two main roles for the variables:

- The *outcome* variable
  - what we are interested in understanding or predicting. Also called dependent or response variable. We only deal with one outcome in this course, however the theory has been extended to multiple outcomes.
- The *predictor* variable(s)
  - what we use to predict or explain the outcome variable. Also called explanatory, independent variable or factor

***Example 1*:**

- *We are given data on 4 different variables for a random sample of 100 staff at the LSE: height (in cm), age (in years), annual income (in 1000's £) and gender (male/female/other)*

What is the aim, prediction or explanation? Which variable is the outcome?

Aim:
Outcome variable:

Type of variable

The type of the variable can also be different.

- *Binary* : these have two values only – 0,1 ; on,off ; Yes,No
- *Categorical* : these have a finite number of values (levels) such that the difference between any two values isn't the same for any pair of values. They are typically qualitative – country, political party, degree. Some have a natural ordering (e.g. degree levels) and could be considered as *Ordinal* but we do not make this difference in the course.

- *Numeric* : these are variables which can take any numerical value. Some common variables are rounded to an integer (e.g. age) or cannot really take on any possible value (% in an exam). Some are discrete integer values (e.g. age in years) others are continuous with fractions or decimal places (e.g. temperature to two decimal places). The key for numeric variables is that the difference between two consecutive values of a variable is the same regardless of which two consecutive values we take.

For *Example 1* identify the type of each variable. Justify your answers.

How are the variables related?

It is very important when encountering a new data set to ask yourself what relationships you *expect to see* between the variables; in particular the outcome and the predictors. You need to think both in terms of sign (+/-) and in terms of strength (is the outcome strongly related to the predictor or is it a weak relationship).

Sometimes you will have no idea! In this case you will hopefully have an expert in the subject matter working with you giving you hints. At other times you will find the sign/importance of a variable diminish when you introduce more variables into a regression. For example, having a bike/car accident may be strongly associated with having severe injuries for the cyclist. If however you include wearing a helmet then this association changes because conditional on wearing a helmet the injuries are less severe.

Thinking about the relationships between variables *before* running the analysis is important for a number of reasons:

- Helps you spot quickly if you need to take a closer look at unexpected results.
- Gives you a deeper understanding of underlying relationships because you have to find ways to explain counter-intuitive results.
- It makes you think more carefully about what the variables are – the unit, the range etc. E.g. if you have data on height and age you might assume that age and height are correlated until you see that the data are for adults only.
- Is helpful when assessing the modelling assumptions (more of that later).

For *Example 1* what do you think the relationships between the outcome and the predictors are? Are any of the predictors likely to be related? Justify your answer.

Group work

In your groups: Identify the outcome and the predictors in the following 2 examples. Try and guess at whether the aim of the research is to explain or predict. Do you anticipate that any of the predictors are associated with one another?

Example 2:

- We are given data on the height (in cms to the shoulder), the breed (Ryeland, Romney, Masham) and weight (in kgs) of wool obtained after shearing of 30 sheep belonging to Farmer Josephine. She has a total of 213 sheep.

What is the aim, prediction or explanation? Which variable is the outcome?

Aim:

Outcome variable:

Identify the type of each variable.

What do you think the relationships between the outcome and the predictors are? Are any of the predictors likely to be related? Justify your answer.

Example 3:

- The data are for a sample of 1000 Londoners who were asked whether they voted (yes/no) in the last general elections, their annual income (in bands 0-20k,20k-30k,30k-40k,40k-50k,50k-60k,60k+), gender (male/female/other), the highest level of education (none/secondary school/university/other) and where in London they live (first two letters of postcode)

What is the aim, prediction or explanation? Which variable is the outcome?

Aim:

Outcome variable:

Identify the type of each variable. Justify your answers.

What do you think the relationships between the outcome and the predictors are? Are any of the predictors likely to be related? Justify your answer.

Plots

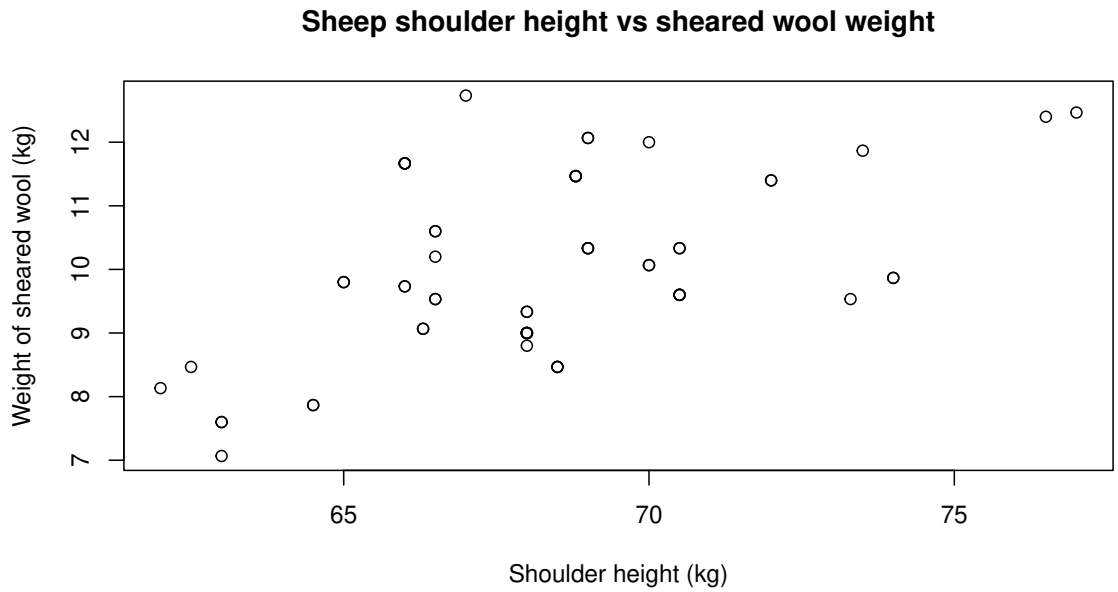
There are two types of plots that you'll use regularly in exploratory analysis: *scatterplots* and *boxplots*. We'll quickly review what they show. As we will use Q-Q plots in residual analysis later on we'll introduce them here.

Scatterplots

Scatterplots should be used to plot *continuous* predictors against *continuous* outcomes. Examples include:

- age (years) vs income (in £)
- skill score (special unit) vs exam grade (in %)
- height (cm) vs income (in £)
- shoulder height (cm) vs weight of wool (in kg) in sheep (see below)

The predictor is in the x-axis and the outcome is in the y-axis.



Scatterplots can be used to visually assess

- whether a linear relationship between predictors and outcome is plausible,
- whether there is any non-linearity (i.e. curvature) in the relationship and
- to get feeling of the strength of the association.

Assess the scatterplot above:

Linearity plausible?:

Non-linearity likely?:

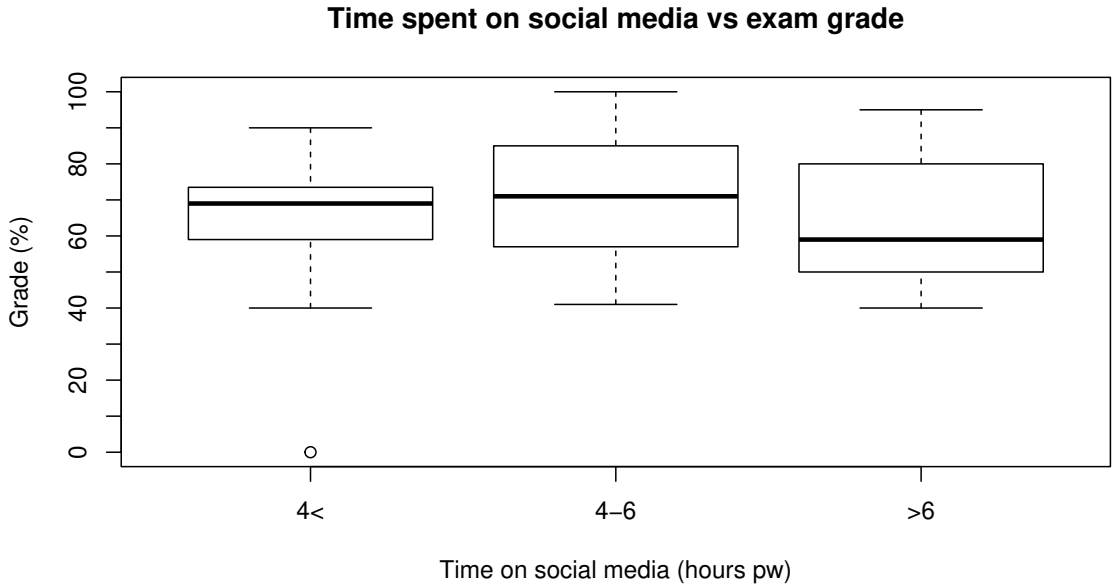
Strong association?:

Boxplots

Boxplots are used to show the relationship between a *categorical* predictor and a *continuous* outcome.

Examples include:

- Gender (male, female) vs income (in £)
- Political affiliation (Tory, Labour, Lib Dem, Other) vs age (in years)
- Time spent on social media (in hour bands per week) vs exam grade (in %)



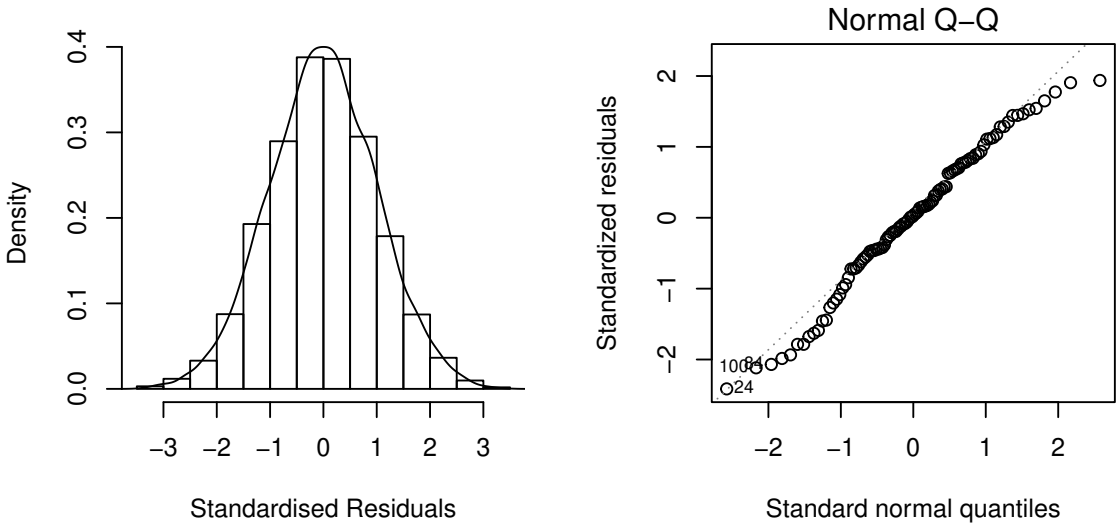
Boxplots can be used to see if there is an association between the outcome and the categorical predictor. We determine this by looking at how different the medians/boxes are from each other.

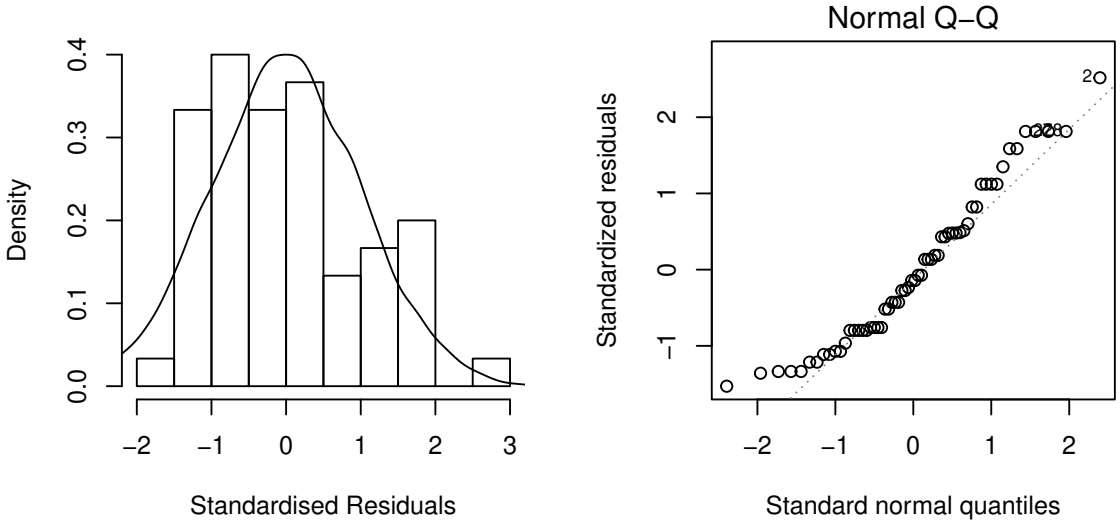
Q: Do you think there is an association between the number of hours spent on social media and exam grades?

Q-Q plots

Q-Q plots are quantile-quantile plots and are closely related to P-P plots (probability plots). We use them in residual analysis in Regression. A Q-Q plot has the quantiles of the standard normal distribution on the x-axis and plots them against the *observed quantiles* (i.e. the quantiles corresponding to the observed histogram) of the data (in our usage the standardised residuals).

> **What is a quantile?** They are points that divide the sample into sections that contain equal probabilities. The median is the central quantile so that half the points in the sample are above it and half the points are below it. The quartiles – 25th, 50th (median) and 75th– divide the sample into 4 sections so that a quarter of the points are below the 25th, half are below the 50th and 3/4 are below the 75th. Wikipedia have a good example on how to determine quartiles (or indeed other quantiles) in their entry on quantiles. For the Q-Q plot n-tiles are created where n is the sample size





- The top LHS is the histogram of points drawn from a standard normal distribution (N(0,1)) with the standard normal density overlayed. We can see that the histogram is perfectly

represented by the density plot.

- The top RHS plot is the corresponding Q-Q plot. It plots the standard normal quantiles against the quantiles of the points from the standard normal (the same ones as in the LHS plot). The dotted line is the x=y diagonal. We can see that the standard normal quantiles and those associated with the sample from the normal are mostly on the x=y diagonal (although sometimes there are some outliers)
- The Q-Q plot assesses how close to normal the standardised residuals are.
- The bottom plots show the standardised residuals from one of our regressions later in the course (I will explain). We can see from both the histogram and the Q-Q plot that these values are not very close to the normal distribution.

As we can see from the plots the histogram has more observations in the low quantiles which corresponds to the larger than standard normal quantile values in the y-axis of the Q-Q plot.

Q: Can you spot any more features in the two top plots that correspond to one another?

Workshop 2: Data Frames

Learning Outcomes

To learn how to load, check correctness of and manipulate data sets in `R`.

Preamble

First of all we need to do the following:

- Open Zoom if you are using it
- Create a new folder (Week 2)
- Open `RStudio` – remember to go for the version with full libraries
- Download the data (on the Moodle page for Week 2) into the folder
- Check it's the right folder (`dir()` lists what is in it)
- If not:
  - Go to the Files tab in the bottom right hand corner of `RStudio`
  - Navigate to the File where you saved the data (e.g. ST211Week2)
  - Click on More at the top of the tab
  - Select "Set as Working Directory" from the drop-down menu

LSE PC alert! Packages

Some of the LSE PCs can't handle loading new packages. It is usually better if you have chosen a version of `RStudio` with the "full libraries" but even that is no guarantee. If your PC is happy with packages then load the "arm" package.

`R` packages

Packages are libraries of functions in `R` code that someone else has designed and coded. In this course we'll use "arm","ggplot2" and some others

Installing `R` packages

For your laptops you need to install the packages first:

- Go to Packages in the bottom right hand corner of `RStudio`
- Click on Install
- Type arm and then click on Install
- Some code will run and arm will appear on the list of User library

Loading `R` packages

To use `R` packages you can either:

Type this in the command line (this is useful to do when you are saving code in a script so that you don't need to manually load packages before running it)

```
library(arm)
```

Or via the `Packages` tab in the bottom right hand corner of `RStudio`. From this tab select the "arm" package from the list of available packages

The "arm" package

The main advantage of "arm" is that is allows us to use the `display()` function with linear models. This has less output than the standard `summary()` command.

If you cannot load "arm" then you can use `summary()` instead of `display()`.

Reading data into R

Data is essential for analysis so learning to load it into R is very important. We load two types of data files into R:

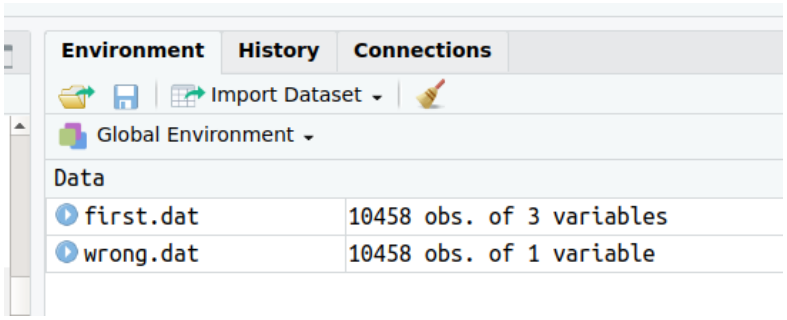
- `.csv` – When a file is a `.csv` file we load it using `read.csv()`.
- `.txt` – For `.txt` file we use `read.table()`.

R can handle many more (e.g. `.csv`, STATA and SAS files) with special packages as well.

We'll start by loading a file of `age` in years by `gender` (male is 1 and female is 0) and hourly pay (`hourpay`) in £ taken from the British Household Panel Survey. The file is in Moodle. *Make sure you download it and save it into your working directory before you try and open it.*

```
first.dat<-read.csv("age_hourpay.csv",header = TRUE)
wrong.dat<-read.table("age_hourpay.csv",header=TRUE)
```

Loading data with the wrong command will result in a nonsense file although not always with an error. You can always look at the Environment tab (top left hand corner of `RStudio`) to make sure. If the length is 1 or there is only 1 variable it is probably wrong.



Things to bear in mind

- The name of the file "age_hourpay.csv" must be in "". This is the *path* that R must take to get to the data file. If the file is not in the folder `RStudio` is running from then you will have to change the path accordingly. (Check using `dir()`)
- `header=TRUE` tells R that the first line of the data should be taken as the *names* of the columns and thus variables in the loaded file. Open your `.csv` file using Excel and you'll see what I mean.
- When R reads a data file using `read.csv` or `read.table` (provided the correct function is used to read it) it creates a `data.frame`. This is the type of data object we will be working with the most and today we will learn to manipulate it.
- We read the data frame into an object(variable) rather than just writing `read.csv("age_hourpay.csv",header = TRUE)` because often our data files are large and then the whole console is taken over by showing the data.
- To see the data, go to the Environment tab in the top left hand corner and click on the grid shape at the end of the line corresponding to the data. The data will appear over the console.

Accessing elements of the data frame

Many of the commands and functions for arrays/vectors work for data frames too. For example:

```
dim(first.dat)
```

```
## [1] 10458     3
```

```
first.dat[55,2]
```

```
## [1] 56
```

Q: What is code to obtain the the age of the 934th individual in the data frame? Bear in mind that age is in the 2nd column.

Some *functions* are for data frames only

```
colnames(first.dat)
```

```
## [1] "gender"  "age"     "hourpay"
```

```
head(first.dat)
```

```
##   gender age hourpay
## 1      0  56  932.50
## 2      0  51  320.51
## 3      0  39  265.40
## 4      0  46  259.28
## 5      0  47  201.93
## 6      0  60  144.25
```

```
head(wrong.dat)
```

```
##   gender.age.hourpay
## 1          0,56,932.5
## 2         0,51,320.51
## 3          0,39,265.4
## 4         0,46,259.28
## 5         0,47,201.93
## 6         0,60,144.25
```

```
tail(first.dat)
```

```
##       gender age hourpay
## 10453      1  39    0.78
## 10454      1  50    0.73
## 10455      0  22    0.60
## 10456      0  29    0.60
## 10457      1  27    0.57
## 10458      1  46    0.47
```

`head()` allows you to look at the first 6 rows of the data and is useful to check if the data are correctly loaded and whether the data makes sense. Whenever calling a large data set (even a large subset of data) it is useful to look at `head()` only as otherwise the data will fill all the console. It is messy and bad practice. If you add a number –n– after the name of the data (separated by a comma) in `head()`you will see the first n rows of that data set. Same with `tail()`.

```
head(first.dat, 3)
```

```
##   gender age hourpay
## 1      0  56  932.50
## 2      0  51  320.51
## 3      0  39  265.40
```

`summary()` gives a summary of the data columns.

```
summary(first.dat)
```

You'll see that the `summary()` function is versatile and can be applied to many `R` objects.

Setting conditions

Let's say we are interested in knowing whether there are people in this data that are below 18.

1. First specify which data set
2. Then specify which variable using the `$` symbol

```
first.dat$age
```

The `$` symbol says – go into `first.dat` and take only the column named `age`. This is a simple but clunky way of accessing the individual columns (variables) in the data. It can also be dangerous as `RStudio` sometimes auto-completes so if you have two variables named "age" and "agent" you may, without realising, end up with very different results.

We could also have written:

```
first.dat[,2]
```

To use this however, you have to know which column corresponds to which variable.

As you can see we've produced lots of data and have maxed out the console. Always use `head()`:

```
head(first.dat[,2])
```

```
## [1] 56 51 39 46 47 60
```

3. Now set your condition < 18.

Start by trying this:

```
head(first.dat$age<18)
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
```

It is not ideal because it goes through first.dat[,2] row by row and answers the question "is this number under 18?". If it is not not the command assigns `FALSE` otherwise `TRUE`.

We usually want to know *which* elements satisfy the condition, i.e. we want a list of positions in the data frame of the elements where the age<18.

4. So which are they?

```
which(first.dat$age<18)
```

This gives their row number in first.dat. We can use this list to see how many there are, create a new data set, look at the values of gender for these, etc.

5. How many are they?

```
length(which(first.dat$age<18))
```

```
## [1] 141
```

6. Let's create a new data.frame with just these people and find out how many women and men there are.

```
under18<-first.dat[which(first.dat$age<18),]
table(under18$gender)
```

```
##
##  0  1
## 57 84
```

Say we want to ignore those below 18 years of age in our analysis. How do we do this? Here are three equivalent expressions

```
over18<-first.dat[which(first.dat$age>=18),]
dim(over18)
```

```
## [1] 10317     3
```

```
over18<-first.dat[first.dat$age>=18,]
dim(over18)
```

```
## [1] 10317     3
```

```
over18<-subset(first.dat, age>=18)
dim(over18)
```

```
## [1] 10317     3
```

You can see that first.dat.over18 now has appeared in Environment. Click on the grid to see the data above the console.

The simplest and most intuitive is the last command using `subset()`. This function takes a subset of the data (the first argument) with criterion the second argument. In this case it takes first.dat and extracts the points where the age is greater or equal to 18. I prefer it to the other expressions as it doesn't have the `$`, the `[]` or the `which()` .

`subset()` can take multiple conditions:

```
second.dat<-subset(first.dat, (age>=18 & hourpay<=50))
```

To do regression we need to use data. We've see how to load data into `R` and how do do some manipulations (e.g. `subset()`) and queries (e.g. `which()`). We've seen how to use the `$` sign to access variables inside data sets, however if we are doing multiple things (such as the previous command) it starts getting really ugly.

`with()` makes things simple and has the same syntax as `subset()`. It takes two arguments.

- the name of the data set in `R`
- the function or operation you want to do to variables in the data.

```
with(first.dat, mean(gender==0))
```

```
## [1] 0.4709313
```

Scatter plots

One of the first things you should do with a new dataset is plot the data in whatever way is possible. For continuous variables a scatter plot is usually used. `plot()` is a very flexible function and traditionally did a lot of the plotting in `R`.

We'll plot age on the x-axis and log(hourpay) on the y-axis. Try plotting age against hourpay to see why we use logs

```
with(over18, plot(age,hourpay))
```

```
with(over18, plot(age,log(hourpay)))
```

"ggplot2" package

For better and more complex plots the best package is the "ggplot2" package. We will be using this most of the time. At the end of your book is a dedicated section on how to use this package with step-by-step instructions. Spend some time going over this as we will be using it a lot.

```
library(ggplot2)
library(gridExtra)
```

The hourly pay vs age plots are shown below with extra features.

```
p1<-ggplot(over18, aes(x=age, y=log(hourpay),colour=factor(gender)))
p1<- p1 + geom_point(size=0.5,aes(colour=factor(gender)))
p1<- p1 +scale_colour_grey(start = 0.1, end = 0.6)  + theme_bw()
p1<- p1+ theme(legend.position = c(0.35,0.9), legend.direction = "horizontal")
p1<- p1+ ggtitle("log(Hourly pay) by age and gender")
p1 <- p1+ geom_smooth(method="lm",fill=NA)


p2<-ggplot(over18, aes(x=age, y=log(hourpay)))
p2<- p2 + geom_jitter(size=0.5,aes(colour=factor(gender)))
p2<-  p2 + scale_colour_grey(start = 0.1, end = .6)  + theme_bw()
p2<- p2+ theme(legend.position = "none")
p2<- p2+ ggtitle("log(Hourly pay) by age and gender")

grid.arrange(p1,p2,ncol=2)
```

Saving data that have been created or changed

It is often the case that we want to change an existing data set or create a new one. In these situations we need to be able to save the data. For example we created the `over18` data from the `first.dat`. Rather than have to re-create it every time we want to use it, we may find it easier to save it directly. There are a number of ways but the one I prefer is using `write.csv`.

```
write.csv(over18, "Over18_age_hourpay.csv")
```

If you run this and get no errors then you should check that there is a new .csv file names Over18_age_hourpay.csv in your working directory.

The arguments of `write.csv` are:

- the name of the `data.frame` to save in `R`
- the name of the .csv file the `data.frame` should be saved into – because it is outside of `R` it needs to be in “”

- `row.names=FALSE` is an optional argument. If it is not included in the call to `write.csv` then it automatically adds a column with the row number to the data. This is very annoying when you later open the dataset and find an additional row.

Try

```
write.csv(over18, "Over18_age_hourpay.csv",row.names=FALSE)
```

and see what the .csv file looks like.

Built in functions

`R` has a large number of built in functions. We've seen a few already

- `log()`
- `plot()`
- `read.csv()`
- `subset()`
- `which()`
- `length()`
- `dim()`

We'll learn many more and also how to write some basic functions.

Remember that functions work as follows: They have a name e.g. `plot()` . Inside the brackets you enter the *arguments* e.g. `x`, `y`, `pch`, `main` etc. Some arguments are mandatory (e.g. `x` and `y`) and if you don't enter them or they are not of the right form then `R` returns an error. Others (e.g. `main`, `pch` etc.) are optional and `R` has some default values.

Other functions useful for stats

- `sum()`
- `mean()`
- `sd()`
- `summary()`
- `exp()`

Try these with our data set.

Doing the same things to each row or column

We often want to do the same thing to every row or column in a data frame. Some things like `colMeans()`,`rowSums()` and `summary()` are built in.

```
summary(over18)
```

```
##      gender            age           hourpay
##  Min.   :0.0000   Min.   :18.00   Min.   :  0.47
##  1st Qu.:0.0000   1st Qu.:32.00   1st Qu.:  7.67
##  Median :1.0000   Median :42.00   Median : 10.94
##  Mean   :0.5282   Mean   :41.84   Mean   : 13.64
##  3rd Qu.:1.0000   3rd Qu.:51.00   3rd Qu.: 16.54
##  Max.   :1.0000   Max.   :65.00   Max.   :932.50
```

```
colMeans(over18)
```

```
##     gender        age    hourpay
##  0.5281574 41.8371620 13.6425007
```

```
head(rowSums(over18))
```

```
##      1      2      3      4      5      6
## 988.50 371.51 304.40 305.28 248.93 204.25
```

`summary()` has a lot of output. Not good if you want to use it in another function or in tailored output

If you are interested you can learn a bit more about functions, loops, the `ifelse()` and `apply()` functions in an excercise in the back of the course notes.

Grouping dataframes

We can use `subset` to group data.

We can use `aggregate` to apply functions to subsets.

```
aggregate(over18[,-1],over18[1],mean)
```

```
##   gender      age  hourpay
## 1      0 42.09922 15.52974
## 2      1 41.60305 11.95649
```

`aggregate` takes the following arguments

- The data to aggregate: in this case the age and hourpay columns
- The column to aggregate them by: in this case the first column of the data
- The function to apply: in this case the mean

As you can see the mean age of women working is lower than that of men but those who do work earn more per hour.

You could try the Loops, functions and apply exercises in the back of the book now

Lecture 3: Simple linear regression

Learning Outcomes

To understand the basic concepts necessary for performing simple linear regression such as lines, least squares estimates, coefficients.

Diagnostics Part 1: Significance of regression coefficients, R^2 statistics and residual standard deviation

Why linear regression?

The aim of linear regression is to fit the relationship between a predictor and an outcome as a *straight line* with coefficients obtained using *least squares estmation*. There are a number of advantages to linear regression:

- lines are easy to understand/interpret.
- lines are easy to fit: least squares regression has easy to calculate formulae for the estimates of the intercept and the slope.
- together with assumptions about the normality of y linear regression can lead to relatively easy checks to ensure that the model is appropriate.
- The theory for Hypothesis tests is straightforward.

How are lines defined in this context:

A straight line on a graph with an x and y axis is given by $y = \beta_0 + \beta_1 x$.

- The line crosses the y axis at β_0: the value for y at $x = 0$ is β_0. β_0 is termed the *intercept*
- The line has a *slope/gradient* of β_1: an increase in x of 1 results in a change of β_1 in y

Terminology

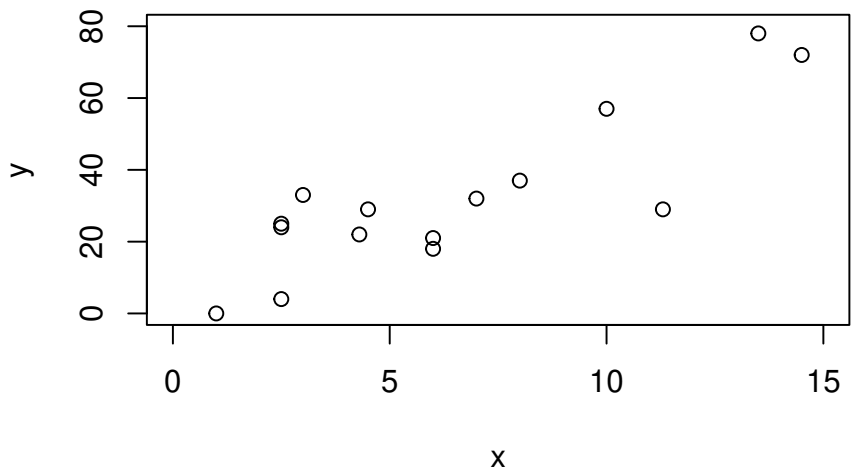
We will use the following terminology

- A predictor is usually termed x (with subscripts if more than one) or given a name e.g. "age"
- The outcome is typically termed y or given a name e.g. "hourpay"
- β is the "true" coefficient, σ is the "true" standard deviation
- Adding a "hat"ˆ indicates an *estimate* rather than the true underlying value (e.g. $\hat{\beta}$ or $\hat{y}_i$)

Estimation

Below is a plot of 15 data points which relate the number of hours of independent study per week = x on the x-axis to the average grade (%) in final exams = y on the y-axis for a subset of LSE students:

```
lse.dat<-read.csv("lsehwk3.csv")
with(lse.dat, plot(x,y, xlim=c(0,15),ylim=c(0,80)))
```



On the plot draw what you think is the line of best fit. Based on this line take a guess at the intercept and the slope. I'm going to guess the intercept it 0 and the slope is 4

$$\hat{y}_i = 0 + 4x_i$$

"Gue-Estimates"

My guesses are (crude) *estimates* of the parameters β_0 and β_1 which we term $\hat{\beta}_0$ and $\hat{\beta}_1$. They are estimates rather than the "true" parameters because

1. our sample is too small (there are only 15 points)
2. we always measure with error (in this case y has been measured using a survey so they are subject to recall bias)
3. are there really any true values? The best possible linear model is at most a good approximation
4. we are guessing (as opposed to estimating the parameters using techniques which have good mathematical and statistical properties :unbiased, converging to the true value etc.)

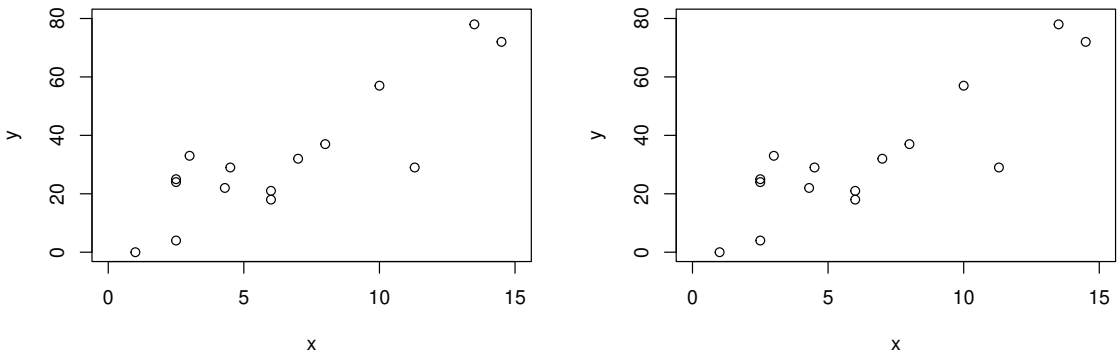
Least squares estimates

Points 1, 2 and 3 are true for least squares estimates too. However they are not a "guess", they are the best estimates possible given the data. Formally, least squares estimates of the coefficients are those that *minimize* the *sum* of the *squares* of the *observed vertical distance* (termed residual) between the points and the line. We could also minimise the sum of the absolute value of the vertical distances. We prefer the *square* because:

- it has better mathematical properties (it is easy to take a derivative of a square)
- it penalises points that are further away more than an absolute value.

In the plot below draw two lines – one that fits better and another that fits poorly and see how that changes the size of the squares of the vertical distance from the line to the point.

```
par(mfrow=c(1,2))
lse.dat<-read.csv("lsehwk3.csv")
with(lse.dat, plot(x,y, xlim=c(0,15),ylim=c(0,80)))
with(lse.dat, plot(x,y, xlim=c(0,15),ylim=c(0,80)))
```



As an aside the least squares line goes through $\bar{x}$ and $\bar{y}$ the means(averages) of x and y respectively

The residuals

The vertical distance between the line of fit and the observed value is called the *residual*. The residuals can be thought of as representing the error in the regression. The bigger they are the worse the fit of the line to the data. Formally:

$$r_i = y_i - \hat{y}_i = y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)$$

or in the case of our guess:

$$\begin{aligned} r_i &= y_i - (\hat{\beta}_0 + \hat{\beta}_1 y_i) \\ &= y_i - (0 + 4x_i) \end{aligned}$$

Sum of Squares of the error (residuals)

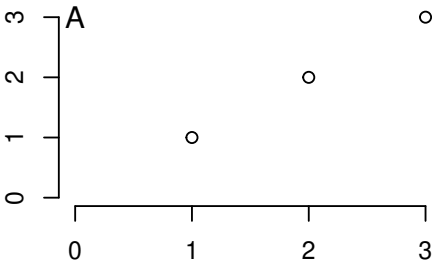
Formally in least squares estimation the quantity that has to me minimised with respect to the parameters β_0 and β_1 is:

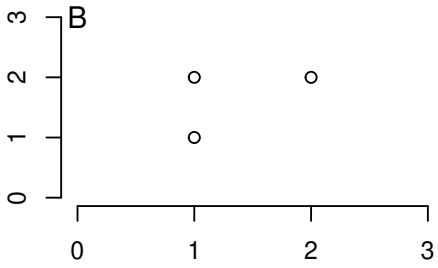
$$\begin{aligned} SSE &= \sum r_i^2 \\ &= \sum [y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)]^2 \end{aligned}$$

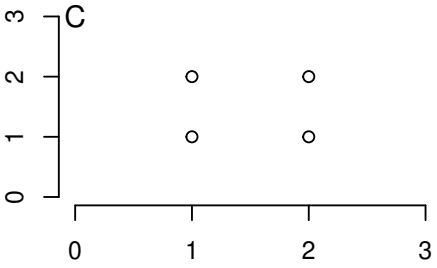
You will find on Moodle a video with the formal derivation of the least squares estimates using calculus. You *MUST* learn this as something related to this is almost always on the exam.

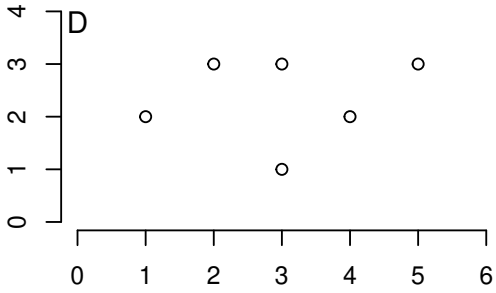
Group work

Below are 4 very simple plots. Draw the line that you think will be best for predicting y from x.









Below are the equations estimated from these data using least squares estimation. Use the four least squares equations to *predict* y for $x = \{0, 3\}$. Then draw a line through the two points in a different colour: this will be your line of best fit using least squares estimation.

- A: $y = 0 + x$
- B: $y = 1 + 0.5x$
- C: $y = 1.5 + 0x$
- D: $y = 2 + 0.1x$

x	A	B	C	D
0				
—	—	—	—	—
3				

Q:Do they look the same?:

Q:If not, why not?

Least squares estimates in R using lm()

Using the data from before let's use R to estimate the regression coefficients. The two functions lm() and display() will become very familiar to you.

lm() stands for *linear model* and it performs a least squares fit. It is versatile and takes many arguments. We use two:

- *the formula*: this takes the form y~x1+x2+x3 where y is the outcome of interest and x1, x2, x3 are predictors. There can be any number of predictors but we start off with one.
- *the data* : this takes the form: data=name.of.dataset where name.of.dataset is the name of the data loaded into R.

For example using the data from the beginning of the lecture:

```
lse.lm<-lm(y~x, data=lse.dat)
display(lse.lm)
```

```
## lm(formula = y ~ x, data = lse.dat)
##             coef.est coef.se
## (Intercept) 3.60     5.71
## x           4.42     0.75
## ---
## n = 15, k = 2
## residual sd = 11.85, R-Squared = 0.73
```

The output of display()

- lm(formula = y ~ x, data = lse.dat): re-iterates the formula
- coef.est coef.se : column headers coefficient estimate and coefficient standard deviation
- (Intercept) : row name for the intercept
- x : row name for the predictor x

- n = 15, k = 2: n= number of data points, k=number of parameters to estimate
- residual sd = 11.85, R-Squared = 0.73: See the section on Diagnostics on the next page.

Writing down a regression

Regression coefficients obtained from least squares or guessing are *estimates* . They are not the *real* values (which probably do not exist as even the best line is an approximation). We acknowledge this in the way we write down regressions.

- The *true* model:

$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$

- Where ϵ_i is the error (Where we assume $\sigma_i = \sigma$ for all $i \in \{1, \ldots, n\}$ and $\epsilon_i \sim N(0, \sigma)$)
- For the observed data I show 2 equivalent expressions of which I prefer the first:
  1. $E(y_i) = \hat{\beta}_0 + \hat{\beta}_1 x_i$
     - E.g:
       $E(y_i) = 3.60 + 4.42 x_i$.
  2. $y_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + r_i$
     - E.g:
       $y_i = 3.60 + 4.42 x_i + r_i$.
- Where r_i is the residual (sometimes ϵ_i is used)
- For $i \in \{1 \ldots n\}$ where n is the sample size
- We usually drop the subscript i

The second formulation explicitly includes the error and says "the observed y is the regression line plus error". The first formulation says "the estimated/expected y is the regression line"

Diagnostics Part 1:

Diagnostic statistics are used to assess how well a model fits the data. They are useful tool for model comparison and fitting, however they should **never** be the only way you assess a model or make decisions about whether to keep a variable in a regression or not. Over the course of this term I will teach you other important approaches.

There are *five* main diagnostics.

1. The standard deviation of the regression line – also known as the *residual standard error*
2. The R^2 (`R-squared/Multiple R-squared`) and
3. The Adjusted R^2 (`Adjusted R-squared`)
4. The significance at the 5% level of the p-value of the t-statistics of the regression coefficients
5. The significance at the 5% level of the p-value of F-statistic of the regression

From `display()` we can get the R^2, the residual standard error and we can calculate the approximate 95% confidence interval which in turn tells us whether the coefficients are significant at the 5% level. We'll discuss the Adjusted R-squared and the F-statistic next week.

Coefficients and their statistical significance

We saw in the first lecture that we can use a t-test to test whether the difference between two means μ_δ is different from 0. We can use the same test for any parameter provided normality assumptions hold. Let us consider the following simple linear regression:

$$\mathrm{E}(y) = \hat{\alpha} + \hat{\beta}x$$

Our main interest is in whether the estimated *slope coefficient* $\hat{\beta}$ is different from 0.

Q: Why is this what we are interested in? What does a 0 slope imply about the relationship between x and y? How well will x predict y if $\beta = 0$?

Formally we test the following:

- *null hypothesis* $H_0 : \beta = 0$
- *alternative hypothesis* $H_1 : \beta \neq 0$

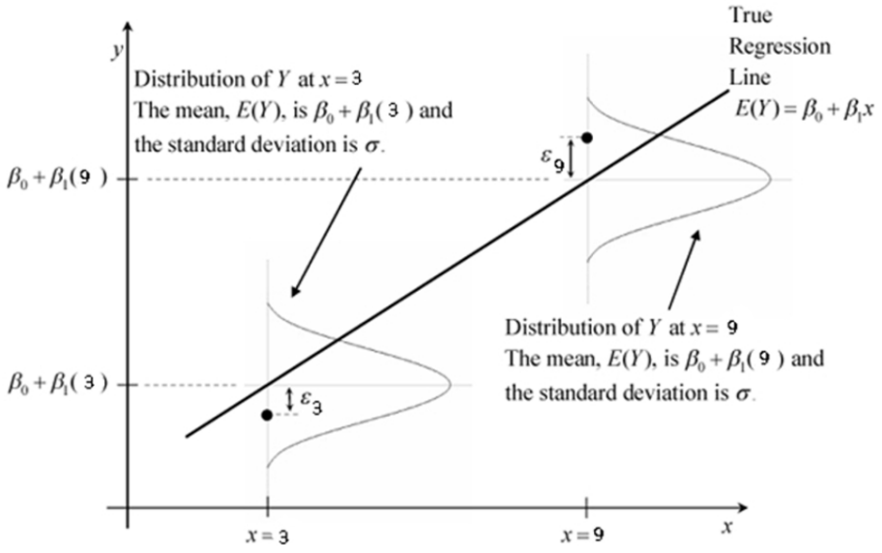
The output of `display()` gives us the standard error of the coefficients/parameters. From our knowledge of the Normal distribution we can obtain an approximate 95% confidence interval for each parameter. If the interval covers the value 0 then we know that the parameter is *non-significant* at the 5% level.

We are typically not that interested in whether the intercept is significant (why?) but we do care about whether the slope coefficients are significant. A poor model would have many non-significant predictors. We always assume we are interested in a 5% level unless otherwise stated.

Q: Based on the output of `display()` above, which, if any of the coefficients are significant at the 5% level? Remember to calculate approximate 95% confidence intervals.

Residual standard deviation

The residual standard error : `residual sd`=$\hat{\sigma}$ is the standard deviation of the estimated regression line. If we look at the output of `display(lse.lm)` and specifically the `residual sd` we see that it has value 11.85. How do we use this information? We tend to say that the smaller $\hat{\sigma}$ is relative to the width of the range of the data (max-min) the better the model fits the data. Why? *If the standard deviation of the line is close to the range of the data, it is as variable as the data and therefore not useful.*



The formula for the residual standard deviation is:

$$\hat{\sigma} = \sqrt{\frac{\sum r_i^2}{n-p}}$$

where n is the sample size and p is the number of predictors + 1, or the number of coefficients estimated by the model. As you can see it is a sort of average of the squared residual – divided by the number of points in the sample minus the number of parameters to be estimated.

Q: What happens to $\hat{\sigma}$ as the number of parameters p increases for fixed sample size n?

Q: What happens as the sample size n increases for a fixed number of parameters p?

The R-squared

The R^2 is a commonly used regression *diagnostic*, i.e. it helps us decide whether the linear model fits the data well.

You could try the Least squares estimate exercises in the back of the book now

Workshop 3: Simple linear regression

Learning outcomes

Perform some exploratory analysis, perform simple linear regressions in `R` and interpret the coefficients.

Basic understanding of the residual assumptions and the residual plots.

Basic understanding of the the R^2 and the residual standard deviation as "goodness of fit" measures.

I will be using both the command `ggplot()` from the "ggplot2" package and `plot()`. Specifically residual plots the `plot()` command in the "base" package in `R` is quicker. For those who have difficulty installing or accessing the "ggplot2" package on your PCs or laptops, I include the code for using `plot()` in some of the plots below but it is generally commented out.

Note: Adding a "#" at the beginning of a line `R` code *comments is out*, i.e. you can see it but `R` ignores

Preamble

Remember to load packages below if you can

```
library(arm)
library(ggplot2)
```

Data

For this and some future classes we'll be using the Stats study habits dataset. This is based on a questionnaire of LSE stats students over the course of 3 years. The questionnaire asked about student's study habits – how often, whether in groups or alone, how they did in tests etc. The responses to some of these questions were then transformed into 8 study skill scores the sum of which we use as a predictor variable here. The outcome of interest is the grade (out of 100) in the stats exam.

Name	Type	Description
Interesting	binary	Is stats interesting? 1 - yes, 0 - no
Study Skills	continuous	0-56 Total study skills - higher is more skills
Grade	continuous	grades in stats exam out of 100
Enjoy	binary	Do you enjoy stats? 1-yes, 0 - no
A-level	continuous	grades in stat/maths A-level out of 7

Let's load the data and check it looks OK.

```
Study_habits<-read.csv("Stats_study_habits_slr.csv")
head(Study_habits)
summary(Study_habits)
```

Q: What is the range of A-level? Study.Skills?

Q: What is the median Grade?

Q: What is the proportion of students who enjoy statistics?

Q: What is the proportion of students who are interested in statistics?

Binary predictors – Interesting

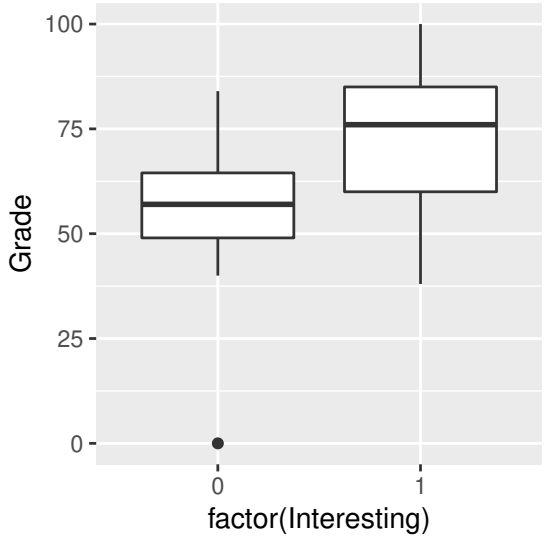
We'll start with one binary predictor: Interesting.

Before we start with the analysis think briefly about what relationship you expect to see between Interesting and Grades. You should get into the habit of doing this. It helps to engage with the data so that you don't blindly accept whatever output R gives you. However you also need to be aware of the fact that relationships in simple linear regression, as expressed by regression coefficients, may change in both size and direction when more than one predictor is introduced.

Q: What do you expect the relationship between Interesting and Grade to be?

Next let's plot the data. Below is a boxplot that shows how the Grades are distributed amongst those who do and don't find stats interesting.

```
#R base package plot
#with(Study_habits,boxplot(Grade~Interesting, xlab="Interesting", ylab="Grade"))
##ggplot2
ggplot(Study_habits, aes(x=factor(Interesting), y=Grade)) + geom_boxplot()
```



Q:What do you infer from the plot?

Let's run a regression using `lm()`

```
grade.interesting.lm<-lm(Grade~Interesting,data=Study_habits)
display(grade.interesting.lm)
```

```
## lm(formula = Grade ~ Interesting, data = Study_habits)
##             coef.est coef.se
## (Intercept) 57.47     2.39
## Interesting 15.26     2.91
## ---
## n = 133, k = 2
## residual sd = 15.68, R-Squared = 0.17
```

Formally: $\mathrm{E}(\mathrm{Grade}) = 57.5 + 15.3 Interesting$

The model tells us about the **difference** in Grades between people who say they find stats interesting vs those who don't.

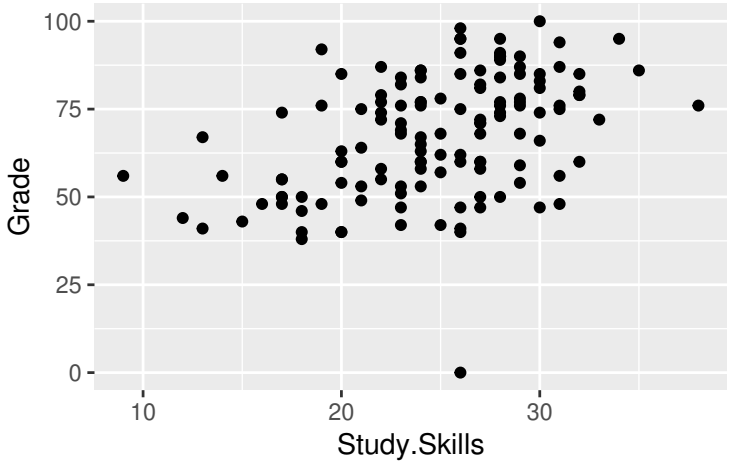
Plug Interesting=0 into the regression and you get the **intercept** 57.5, i.e. the predicted Grade for students who do not find statistics interesting. Plug Interesting=1 into the equation you get $57.5 + 15.3 = 72.8$ which is the predicted Grade for students who find statistics interesting. 15.3 is the **slope** and tells us that students who find statistics interesting have **on average** a 15.3 higher Grade than students who do not.

Continuous predictor – Study.Skills

We now use a single continuous predictor: Study.Skills. Study.Skills is the sum of the scores obtained by students for a number of study skill related variables. The higher it is the more study skills the student has. The maximum possible is 56 although no-one in the sample attained that.

Q:What do you anticipate the relationship between Study.Skills and Grades to be?

```
#R base package
#with(Study_habits, plot(Study.Skills,Grade,pch=4))
ggplot(Study_habits, aes(x=Study.Skills, y=Grade))+ geom_point()
```



Q: What do you infer from the plot?

The "ggplot2" package is part of the "tidyverse" approach to coding in `R` which is becoming increasingly popular. Instead of having a function and adding all the arguments inside it, it has a core command `ggplot()` which defines the data and the variables (`x,y,fill,colour`) and then additional function follow with a `+`

We've seen that in both the cases above the `ggplot()` command was similar, with just the `x` changing value. The first plot was a boxplot so `+ geom_boxplot()` was added. The second plot was a scatterplot so `+ geom_point()` was added.

In order to make sure the whole line of code is visible in the output on the page in front of you, from now on, I assign a name (`p1`) to the plot I create in "ggplot()" and then add arguments to `p1` in subsequent lines.

Now for the regression:

```
grade.studyskills.lm<-lm(Grade~Study.Skills, data=Study_habits)
display(grade.studyskills.lm)
```

```
## lm(formula = Grade ~ Study.Skills, data = Study_habits)
##              coef.est coef.se
## (Intercept)  30.77     6.70
## Study.Skills  1.49     0.26
## ---
## n = 133, k = 2
## residual sd = 15.47, R-Squared = 0.20
```

Formally $\mathrm{E}(\mathrm{Grade}) = 38.8 + 1.5 Study.Skills$

The model tells us by how much the *average* Grade changes when Study.Skills increases by *one*.

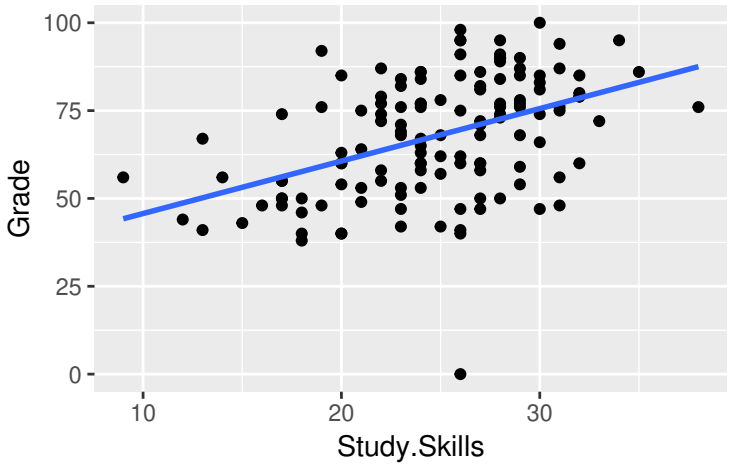
Plug Study.Skills=0 into the regression equation, this gives you the **intercept**: 38.8 which is the predicted score for students who have a 0 Study.Skills score. $30.8 + 1.5 = 32.3$ is the predicted score for students who have value 1 Study.Skills score. $30.8 + 1.5 \times 10 = 45.8$ is the predicted score for a student with Study.Skills score of 10. 1.5 is the **slope** and tells us that the predicted Grade increases **on average** by 1.5 for an increase of 1 in the study skills score.

Q: Are any students likely to have a Study skills score of 0?

Q: If not, how is the intercept to be interpreted?

Adding a regression line to the plot

```
p1<- ggplot(Study_habits, aes(x=Study.Skills, y=Grade))+ geom_point()
p1<- p1 + geom_smooth(method="lm",fill=NA)
p1
```



If not using the "ggplot2" package, you can use the `abline()` function to do something similar. It can be used in 3 modes:
1. Adding a regression line `abline(grade.studyskills.lm)`
2. Adding a line using the intercept (`a`) and the slope (`b`)
`abline(a=40,b=1.5, lty=2)`
3. Adding vertical or horizontal lines `abline(v=20,lty=3), abline(h=60,lty=4)`

`lty` is a graphical parameter that specifies the line type (dashed=2, solid=1). If you Google "lty R" you'll see a list.

```
#with(Study_habits, plot(Study.Skills,Grade,pch=4))
#abline(grade.studyskills.lm)
#abline(a=40,b=1.5, lty=2)
#abline(v=mean(Study_habits$Study.Skills),lty=3)
#abline(h=mean(Study_habits$Grade),lty=4)
```

Residual Assumptions

The assumptions we discuss below are about the error in a regression, that is, the difference between the "true" regression line and the observed points. However we don't observe the error directly because all we have is the estimated line so we use the residuals as a proxy. Note that assumptions about the error are mirrored by assumptions about the outcome as the residual is the outcome minus a straight line (i.e. the residual is a linear transformation of the outcome).

Normality

Hypothesis tests all rely on the errors (ϵ) being approximately normally distributed in linear regression. If normality does not hold then the hypothesis tests can be invalid.

Predictions also become unreliable if we cannot rely on normality.

It is always important to check *approximate* normality of residuals. Linear regression is still valuable when the residuals aren't normally distributed, but results are less reliable.

Normality is checked using:

- The histogram of the residuals.
- The Q-Q plot.

Independence

The error terms have to be independent of one another. Consider the following two situations:

1. Data are daily temperatures measured outside Columbia house in the MT 2018

2. Data are number of beers sold from a beach shack in Australia and the number of shark attacks in the area over 10 years

1. Temperatures

If we regress temperature on day from the first to the last day of the MT 2018 the residuals will be correlated.

Q: Why are the residuals correlated?

We don't look at time series in this course so we will not come across this type of problem directly. However independence of this type can be checked by plotting the errors against time – you will expect to see some sort of seasonal pattern.

2. Beer/Sharks

If we regress shark attacks per summer on number of beers sold we see that the two are correlated. However the regression will also result in correlated errors.

Q: Why are the residuals correlated?

This type of problem can sometimes be avoided by trying to understand how variables in a data set are related and whether there are any predictor variables that might take the role of *confounding* variables. I.e. variables that are somehow (causally?) related to the other predictors such that omitting them introduces dependence and therefore bias into the regression.

If we include data on them in the regression we may be able to overcome the problem. Including these variables can even result in a change of sign! Other times we simply do not have data on these variables and must accept that our analysis is likely to be incorrect to some extent. These variables are also called, lurking, omitted. There is an exercise in the back of the course notes to investigate the effect of confounding on estimates of regression coefficients.

Constant Variance

In order for linear regression to be useful we must assume that the errors have a common variance – i.e. there is no *heteroscedasticity*. If this assumption is violated then our estimates for standard errors become unreliable as do Hypothesis tests.

Often when there is lack of common variance this can be addressed either by transforming the data or by using non-linear models. We check heteroscedasticity by using:

- The plot of the fitted values vs the residuals
  - if this displays a funnel shape
  - if this displays a non-linear relationship

then there is evidence of heteroscedasticity.

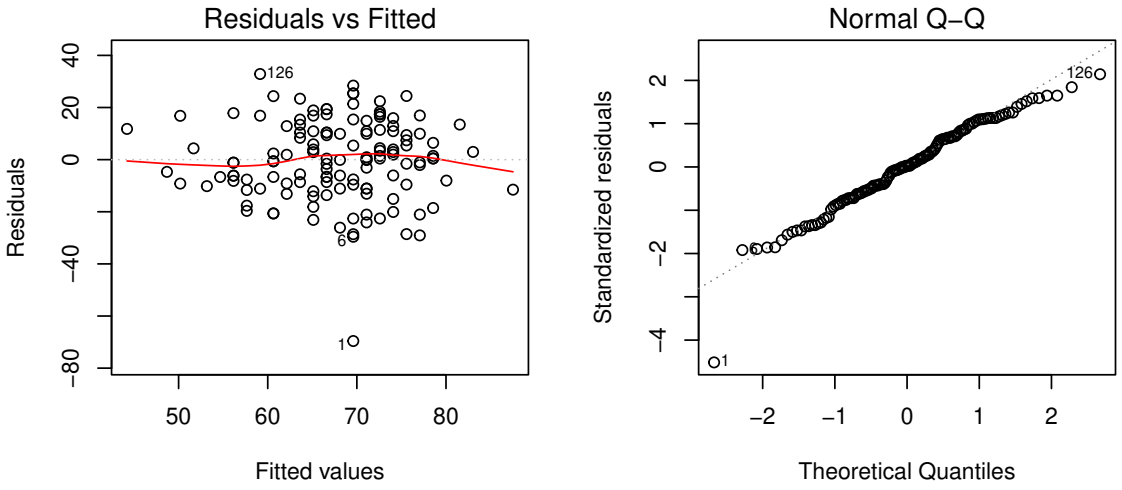
Zero expectation

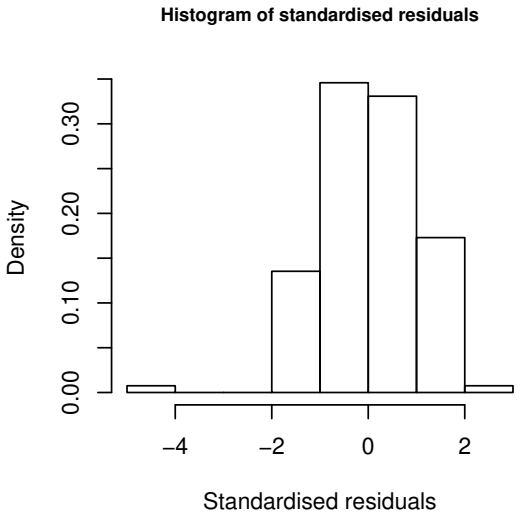
The error terms should have a 0 mean. This assumption holds approximately with least squares estimation if the relationships are linear (it's designed that way). If relationships are non-linear then it won't hold. In practice the mean of the residuals will never be exactly 0, however it will be as small as possible.

Residual plots

The plots used to check the residual assumptions are called *residual plots*. We want 3 plots so we ask for a 2 by 2 grid using `par(mfrow=c(r,c))` (3 in a row or in a column makes plots look squashed or takes up too much room). For residual plots we use the `plot()` function as it easily produces residual plots when it is passed a linear model.

```
par(mfrow=c(2,2))
plot(grade.studyskills.lm, which=c(1,2))
hist(rstandard(grade.studyskills.lm), freq = FALSE ,
     main="Histogram of standardised residuals",
     cex.main=0.8, xlab="Standardised residuals")
```





Let's go through this code:

- `plot(grade.studyskills.lm)` produces 4 plots. We only want 2 of them, the residual vs fitted and the Q-Q plot
- As these are the 1st two we add an argument `which=c(1,2)`
- `hist(x)` produces a histogram of x
- `rstandard(grade.studyskills.lm)` are the standardised residuals
- `main="Histogram of residuals"` is the title of the plot – Histogram of residuals
- `font.main=1` determines the size of the font of the title
- `xlab="Residuals"` says that the x label – Residuals
- `probability=TRUE` gives the probability histogram rather than the frequency histogram

Group work

For the three residual plots above, do you think the residual assumptions hold? If not, why not?

Normality:

Constant variance:

Some goodness of fit statistics: Residual standard deviation and R-squared

Let's have another look at the output from `display`:

```
grade.lm.0<-lm(Grade~Study.Skills, data=Study_habits)
display(grade.lm.0)
```

```
## lm(formula = Grade ~ Study.Skills, data = Study_habits)
##              coef.est coef.se
## (Intercept)  30.77     6.70
## Study.Skills  1.49     0.26
## ---
## n = 133, k = 2
## residual sd = 15.47, R-Squared = 0.20
```

At the bottom of the output are two statistics: the residual standard deviation (`residual sd`) and the R^2 (`R-Squared`). These statistics are crude (but often effective) measures of how well the model fits and/or explains the data.

The residual standard deviation is the standard deviation of the estimated regression line. It should be smaller than the range of the outcome:

In our case max(Grade)-min(Grade)=100 and $\hat{\sigma}$=15.47

Q: Interpret the residual standard deviation in this example:

The lower the R-squared the better the fit of the line. In our case it is 0.20 so 20%.

Q: Comment on the R-squared in this example:

For the `plot()` command only:

> Often you will want to put more than one graph on the same page. If you use `plot()` then you'll need to put `par(mfrow=c(num.rows,num.columns))` before the plots.
> Note that if you close the plot then the `par()` option gets forgotten.
> For example:

```
par(mfrow=c(2,1))
plot(seq(1,10),seq(1,10))
plot(seq(1,10),seq(1,10),col="red")
```

Then try:

```
par(mfrow=c(1,2))
plot(seq(1,10),seq(1,10))
plot(seq(1,10),seq(1,10),col="red")
```

Lecture 4: Introduction to Multiple Linear Regression

Learning outcomes:

Extending linear regression to include one binary and one continuous predictor

Understanding interactions between a binary and a continuous predictor in multiple linear regression

Data

In this lecture we'll use the Study Skills data set again but we'll use *two* predictors. `Interesting` and `Study.Skills`.

Name	Type	Description
Interesting	binary	Is stats interesting? 1 - yes , 0 - no
Study Skills	continuous	0-56 Total study skills - higher is more skills
Grade	continuous	grades in stats exam out of 100

```
Study_habits<-read.csv("Stats_study_habits_mlr.csv")
head(Study_habits)
```

With two predictors the "true" regression looks like this:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Using our data this becomes:

$$Grade = \beta_0 + \beta_1 Study.Skills + \beta_2 Interesting$$

Q: Before running the regression, what do you think the signs of β_1 and β_2 are?

Q: Do you think that Study.Skills and Interesting *interact*? I.e. do you think that students who are interested benefit more/less from having higher study skills that students who are not interested? By "benefit" I mean "get higher grades".

This is a complex question and I just want you to bear it in mind when we look at subsets regression and interactions later in the lecture.

Let's run the regression. Below are the values for the regression coefficients:

```
grade.lm.1<-lm(Grade~Study.Skills+Interesting,data=Study_habits)
coef(grade.lm.1)
```

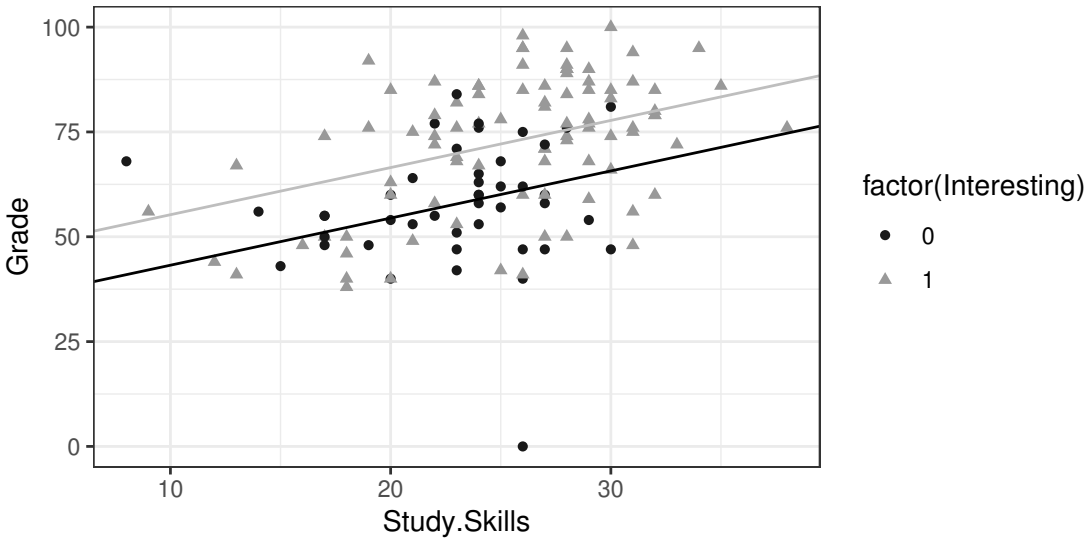
```
##  (Intercept) Study.Skills  Interesting
##    31.975960     1.124188    12.041984
```

The regression looks like this:

$$E(Grade) = 31.98 + 1.12 Study.Skills + 12.04 Interesting$$

Plots for multiple predictors

We saw how to plot this data for one predictor. Below is how it can be plotted for a continuous and a binary (or categorical) predictor.



Q: What do you notice about the slopes of the two lines?

The regression line above is using the value of interesting to change the intercept only. So, when `Interesting`=0 the line is

$$E(Grade) = 31.98 + 1.12 Study.Skills$$

But when `Interesting`=1 the line is

$$E(Grade) = 31.98 + 1.12 Study.Skills + 12.04$$

$$= 44.02 + 1.12 Study.Skills$$

Q: Do you think that imposing the same slope is sensible? Justify your answer based on the data/knowledge

Interpreting coefficients

Looking at the regression again:

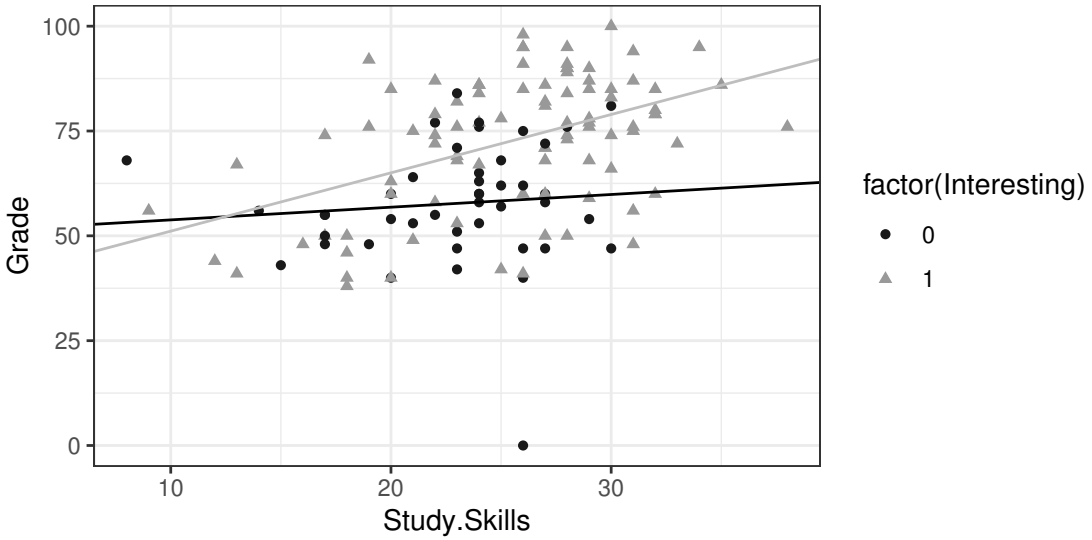
$$E(Grade) = 31.98 + 1.12 Study.Skills + 12.04 Interesting$$

- The *intercept*: A students with no study skills (`Study.Skills`=0) and with no interest in statistics (`Interesting`=0) has an average predicted `Grade` of 31.98. Again not very useful as `Study.Skills`=0 is unlikely.
- The *coefficient* of `Interesting`: When we look at students who have *the same* level of `Study.Skills` then those who are interested in statistics are on average getting 12.04 more in their `Grade` than those who are not interested.
- The *coefficient* of `Study.Skills`: We expect students to get on average 1.12 more in their `Grade` for every additional point in `Study.Skills` score for any level of `Interesting`

The basic idea behind interpreting coefficients is thinking how changing one predictor changes the average outcome *while keeping the values/levels of the other predictors the* **same**.

Subset Regression

We can explore whether the same slope assumption is reasonable by running two separate regressions: one for those who are interested and another for those who are not:



Q: What do you notice about the lines in this plot as compared to the one above?

From the plot we can see that the slope for the students who are not interested is approximately 0. That means that having a higher study skills score doesn't help them to get better grades in their statistics exam. Students who are interested however do benefit from having a higher study skills score as this slope is positive.

So should we always do subset regression?

- No. You *should* do if when the interest is in specific subsets/sub populations (e.g. countries, regions)
- Other times it is not necessary. For example when there are few subsets (e.g. men vs women)
- Or not every subset is relevant or important
- Often including an *interaction* takes into account differences between subsets.

Interactions

As we saw from the subset regression the slopes of the regression lines for the interested vs uninterested students are quite different. This suggest that being interested has a different effect on

the grades of those who are interested and those who are un-interested. We can incorporate this into a single regression using an interaction term: The interaction is the *product* of the predictors:

$$Grade = \beta_0 + \beta_1 Study.Skills + \beta_2 Interesting + \beta_3 Study.Skills \times Interesting$$

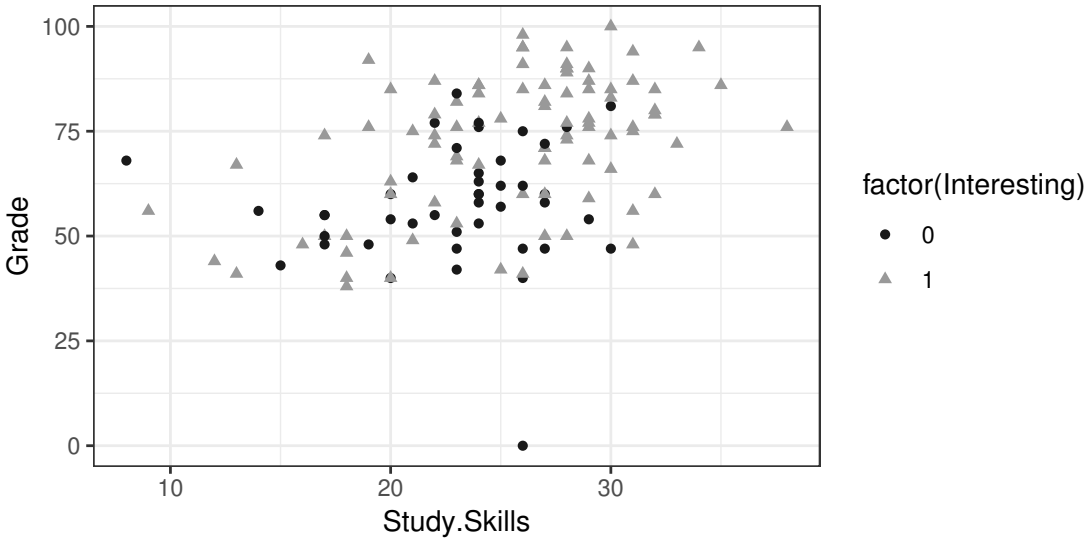
```
grade.lm.int<-lm(Grade~Study.Skills*Interesting,data=Study_habits)
coef(grade.lm.int)
```

```
##               (Intercept)              Study.Skills               Interesting
##                50.7798909                 0.3025668               -13.5712854
## Study.Skills:Interesting
##                 1.0883059
```

Group work

$$E(Grade) = 50.78 + 0.3 Study.Skills - 13.57 Interesting + 1.09 Study.Skills \times Interesting$$

Q: Using the regression equation above write down two regression lines. One for students who are interested in statistics and one for students who are not interested. Once you have done this, plot the slopes on the scatterplot below.



Before, `Interesting` only contributed to changing the intercept. Now it also contributes to changing the slope.

Interpretation with an interaction

Once we've run a regression with an interaction it is not easy to interpret the parameters *on their own*. The best thing to do is to write down the separate regressions. This is of course more difficult and complex the more interaction terms there are.

However, if you want to try to interpret then:

The *intercept* and the *coefficient* of `Interesting` alone are not easy to interpret as both require that the value of `Study.Skills`=0, which as we said is unlikely.

- The coefficient of `Study.Skills` can be thought of as a comparison between the average `Grade` for students who are not interested when their study skill score increases by 1.
- The coefficient of the interaction is the change in the coefficient (slope) of `Study.Skills` between students who aren't interested and those who are.

When to add interactions

- Sometimes we are told by subject matter experts that interactions should be included. For example in medical data sex/gender are always included in the model and so are their interactions with other predictors.
- When one or more of the main effects are large, it is a good idea to include interactions at least between the large main effects.
- When some subsets of the data are considered of particular interest (although sometimes subsets regression is better for interpretation if there are many groups – especially if results need to be explained to subject matter experts who may not be great with equations)
- *Note*: Interactions between two continuous predictors are somewhat harder to understand. They also often lead to multicollinearity and overfitting (discussed later). I would avoid them unless there are strong substantive reasons to add them. In that case there are some interesting ways of visualising them in plots.

Workshop 4: Multiple Linear Regression

Learning outcomes

Running a MLR with multiple predictors

Understanding how to plot this to see interactions

Diagnostics Part 2: Adjusted R^2 and F-statistics

In your own time: `ifelse()`

Preamble

Load the following packages:

```
library(arm)
library(ggplot2)
```

Data

In this workshop we'll use the Study Skills data set again but we'll consider the predictors `Enjoy` and `Study.Skills`.

Name	Type	Description
Enjoy	binary	Enjoy, 1 - yes, 0 - no
Study Skills	continuous	0-56 Total study skills - higher is more skills
Grade	continuous	grades in stats exam out of 100

```
Study_habits<-read.csv("Stats_study_habits_mlr.csv")
head(Study_habits, 4)
```

```
##   Interesting Enjoy A.Level Study.Skills Grade
## 1           1     1       6           18    38
## 2           1     0       6           18    40
## 3           0     0       6           20    40
## 4           1     0       6           20    40
```

The equation is:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Or in our case

$$Grade = \beta_0 + \beta_1 Study.Skills + \beta_2 Enjoy$$

Q: Before running the regression, what do you think the signs of β_1 and β_2 are?

Q: Do you think that "Study.Skills" and "Enjoy" *interact*? I.e. do you think that students who enjoy statistics get higher(lower) grades by having higher study skills that students who do not enjoy statistics?

Let's run the regression to see what the least squares *estimates* $\hat{\beta}$ are.

```
grade.lm.1<-lm(Grade~Study.Skills+Enjoy,data=Study_habits)
display(grade.lm.1)
```

```
## lm(formula = Grade ~ Study.Skills + Enjoy, data = Study_habits)
##              coef.est coef.se
## (Intercept)  32.51     6.51
## Study.Skills  1.39     0.26
## Enjoy         4.10     3.18
## ---
## n = 134, k = 3
## residual sd = 15.52, R-Squared = 0.19
```

The function `coef()` returns a *vector* of coefficients. If we are just interested in their value this is a useful function. The first is always the intercept (unless a no intercept model has been specified) the rest are in the order in which they appear in the call. So for us `Study.Skills` first followed by `Enjoy`

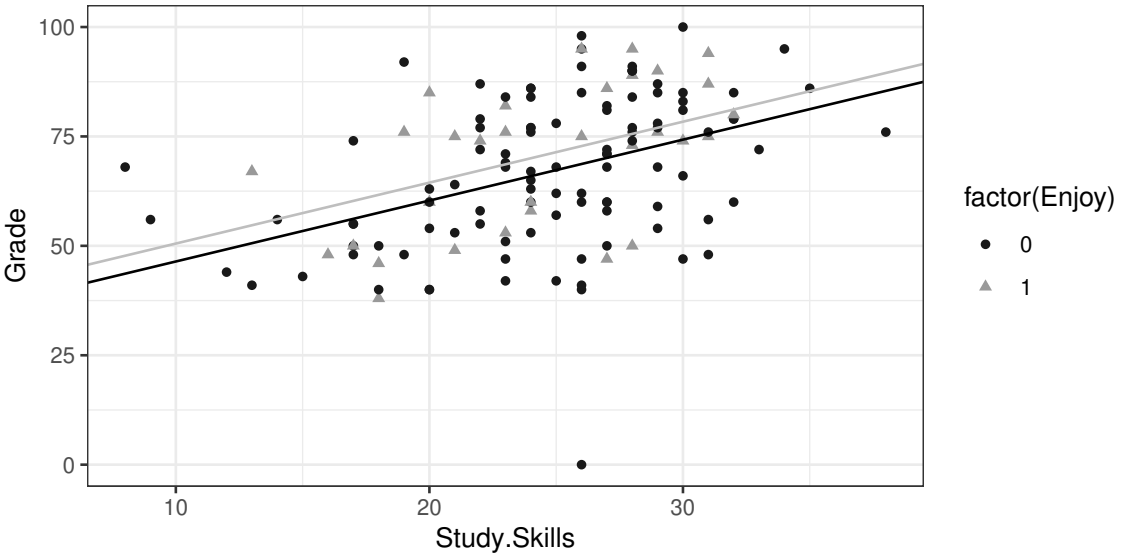
```
coef(grade.lm.1)
```

```
##  (Intercept) Study.Skills        Enjoy
##    32.509625     1.392049     4.096415
```

Plots for multiple predictors

As we saw in the lectures it is often useful to produce plots that allow us to visualise the regression lines for different levels of a binary/categorical variable. Below is the plot corresponding to no-interaction. On the next page is the `R` code using `ggplot()`. I will go over it carefully, there is space on the next page to write notes to help you understand what is going on.

Q: Based on the plot below do you think that the effect of "Enjoy" is very big? Justify your answer



```r
# plotting the points and lines set up the plot:
p1 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, colour = factor(Enjoy)))
# size and colour of the points by Enjoy
p1 <- p1 + geom_point(aes(shape = factor(Enjoy)))
# Note: it's easier in ggplot to plot lines with interactions! Plot the line
# with intercept and slope only for Enjoy=0
p1 <- p1 + geom_abline(intercept = coef(grade.lm.1)[1], slope = coef(grade.lm.1)[2],
    color = "black")
# Plot the line with intercept = intercept+coefficient of Enjoy and slope
# for Enjoy=1
p1 <- p1 + geom_abline(intercept = (coef(grade.lm.1)[1] + coef(grade.lm.1)[3]),
    slope = coef(grade.lm.1)[2], color = "gray")
# make it black and white (ignore for now but useful for black and white
# printing)
p1 <- p1 + scale_colour_grey(start = 0.1, end = 0.6) + theme_bw()
p1
```

Use this section to write notes to help you understand the code.

As we saw in the lectures, when we don't include an interaction between `Enjoy` and `Study.Skills` the lines are parallel. This is because the line is using `Enjoy` to change the intercept only. The regression looks like this:

$$E(Grade) = 32.51 + 1.39 Study.Skills + 4.1 Enjoy$$

So, when `Enjoy`=0 the line is:

$$E(Grade) = 32.51 + 1.39 Study.Skills$$

But when `Enjoy`=1 the line is

$$\begin{aligned} E(Grade) &= 32.51 + 1.39 Study.Skills + 4.1 \\ &= 36.61 + 1.39 Study.Skills \end{aligned}$$

Interpreting coefficients

- The *intercept*: A students with no study skills (`Study.Skills`=0) and who does not enjoy statistics (`Enjoy`=0) has a predicted Grade of 32.51. Again not very useful as `Study.Skills`=0 is unlikely.
- The *coefficient* of `Enjoy`: When we look at students who have *the same* level of `Study.Skills` then those who Enjoy statistics are on average getting 4.1 more in their Grade than those who do not Enjoy.
- The *coefficient* of `Study.Skills`: When we look at students who enjoy statistics we expect them to get on average 1.39 more in their grade for every additional point in study skills score. Same for students who do not enjoy statistics.

The basic idea behind interpreting coefficients is thinking how changing one predictor changes the average outcome *while keeping everthing else the* **same**.

Subset regression

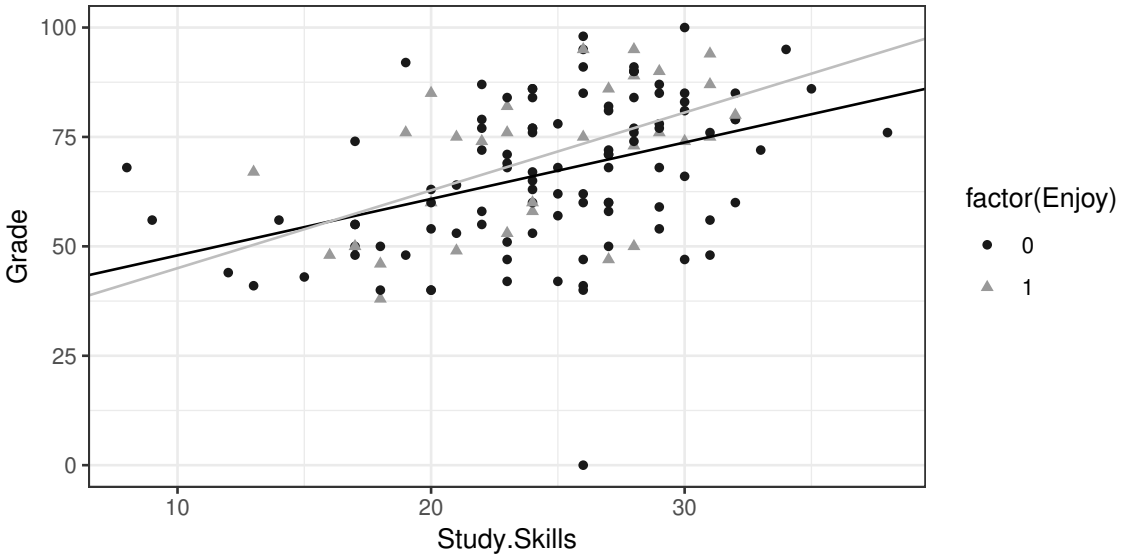
Let's divide the data into two groups, those who enjoy statistics and those who don't and run separate regressions.

First we create two new subsets:

```
#Subset the data and then run the regression of Grade on Study.Skills for each subset
Study_habits.Enj1<-subset(Study_habits, Enjoy==1)
lm.Enj1<-lm(Grade~Study.Skills,data=Study_habits.Enj1)
Study_habits.Enj0<-subset(Study_habits, Enjoy==0)
lm.Enj0<-lm(Grade~Study.Skills,data=Study_habits.Enj0)
```

Then create two separate slopes on the same plot:

```
# Basic plot p1 of Grade against Study.Skills distingushing points by Enjoy
p1 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, colour = factor(Enjoy)))
p1 <- p1 + geom_point(aes(shape = factor(Enjoy)))
# Plot a line based on the lm.Int0 for the subset of people with Enjoy=0
p1 <- p1 + geom_abline(intercept = coef(lm.Enj0)[1], slope = coef(lm.Enj0)[2],
    color = "black")
# Plot a line based on the lm.Int1 for the subset of people with Enjoy=1
p1 <- p1 + geom_abline(intercept = coef(lm.Enj1)[1], slope = coef(lm.Enj1)[2],
    color = "gray")
# for black and white. you can ignore
p1 <- p1 + scale_colour_grey(start = 0.1, end = 0.6) + theme_bw()
p1
```



Interactions

From the scatterplot above and from the regression, the effect of `Enjoy` does not appear to be very strong, however there is no way of knowing this for sure until we look at the formal diagnostics later. So let's try adding an interaction.

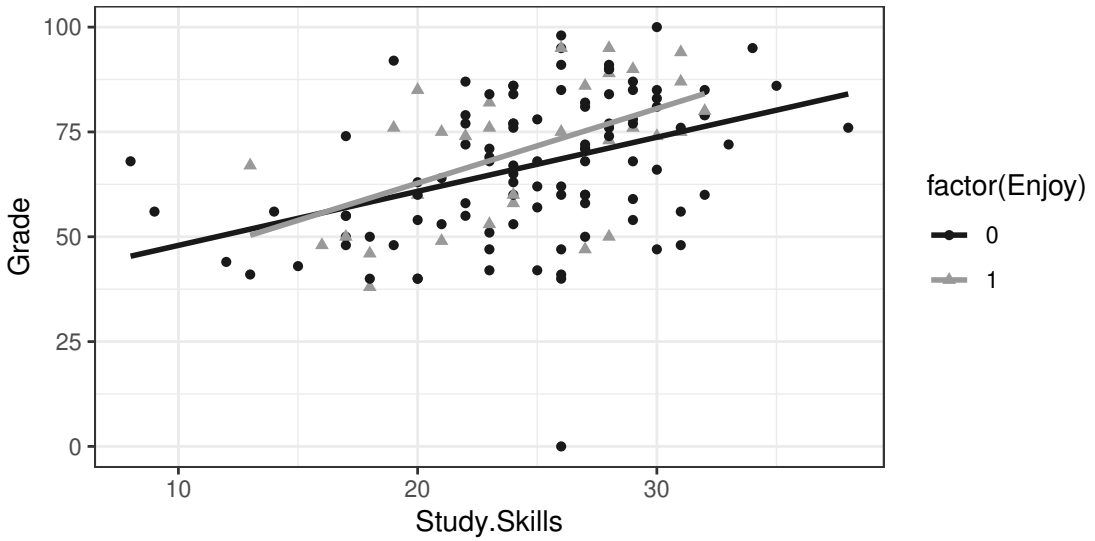
```
grade.enjoy.lm.int<-lm(Grade~Study.Skills*Enjoy,data=Study_habits)
display(grade.enjoy.lm.int)
```

```
## lm(formula = Grade ~ Study.Skills * Enjoy, data = Study_habits)
##                     coef.est coef.se
## (Intercept)         35.04     7.29
## Study.Skills         1.29     0.29
## Enjoy               -7.81    15.67
## Study.Skills:Enjoy   0.49     0.63
## ---
## n = 134, k = 4
## residual sd = 15.54, R-Squared = 0.19
```

Q: Write down two regressions, one for the case where `Enjoy` is 0 and another for `Enjoy` is 1.

The code for interactions in the `ggplot()` it much easier as you can see:

```
#Basic plot
p1<-ggplot(Study_habits, aes(x=Study.Skills, y=Grade,colour=factor(Enjoy)))
#add points
p1<- p1 + geom_point(aes(shape=factor(Enjoy)))
#In ggplot it is much easier to plot the interaction!
# se=FALSE doesn't add the standard error range to the regression. Try without it!
p1<- p1+geom_smooth(method="lm", se=FALSE)
#black and white colour scheme, you can ignore
p1<- p1 +scale_colour_grey(start = 0.1, end = 0.6)  + theme_bw()
p1
```



Q: Based on the plot and the output from the regression, do you think `Enjoy` makes a difference to the average `Grade`? Does it interact with `Study.Skills`? Justify your answer.

Diagnostics Part 2

Diagnostic statistics are used to assess how well a model fits the data. They are useful tool for model comparison and fitting, however they should **never** be the only way you assess a model or make decisions about whether to keep a variable in a regression or not. Over the course of this term I will teach you other important approaches.

A QUICK GUIDE TO MODEL ASSESSMENT can be found at the back of the book. This will be useful for the Reading week project and is the *minimum* you need to do in any analysis.

Remember that there are *five* main diagnostics. We're familiar with some of them:

1. The standard deviation of the regression line – also known as the *residual standard error*
2. The R^2 (`R-squared/Multiple R-squared`) and
3. The Adjusted R^2 (`Adjusted R-squared`)
4. The significance at the 5% level of the p-value of the t-statistics of the regression coefficients
5. The significance at the 5% level of the p-value of F-statistic of the regression

From `display()` we can get the R^2, the residual standard error and we can calculate the approximate 95% confidence interval which in turn tells us whether the coefficients are significant at the 5% level. Let's add all the predictors we have considered to be important to the regression. We also remove the Grade=0 point.

```
Study_Habits<-read.csv("Stats_study_habits_noout.csv",header=T)
grade.lm<-lm(Grade~Study.Skills+Interesting+Enjoy,data=Study_Habits)
display(grade.lm)
```

```
## lm(formula = Grade ~ Study.Skills + Interesting + Enjoy, data = Study_Habits)
##              coef.est coef.se
## (Intercept)  24.24     7.48
## Study.Skills  1.41     0.32
## Interesting  10.10     3.52
## Enjoy         3.25     3.43
## ---
## n = 85, k = 4
## residual sd = 13.80, R-Squared = 0.35
```

We cannot calculate the F-statistic or the Adjusted R^2 from this. In order to get all these values we use `summary()`

```
summary(grade.lm)
```

```
##
## Call:
## lm(formula = Grade ~ Study.Skills + Interesting + Enjoy, data = Study_Habits)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -29.9641  -8.7788  -0.7915  10.2085  27.3938
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   24.2370     7.4787   3.241  0.00173 **
## Study.Skills   1.4074     0.3154   4.462 2.59e-05 ***
## Interesting   10.0990     3.5246   2.865  0.00531 **
## Enjoy          3.2500     3.4320   0.947  0.34647
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.8 on 81 degrees of freedom
```

```
## Multiple R-squared:  0.352,  Adjusted R-squared:  0.328
## F-statistic: 14.67 on 3 and 81 DF,  p-value: 1.026e-07
```

F-statistic

The F-statistic is a measure of how good the model *as a whole* is. We are usually interested in the p-value associated with it. Let us consider a regression given by:

$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots \beta_p x_{pi}$

Then, for the F-test,

- The *null hypothesis* , $H_0 :\ \beta_1 = \beta_2 = \ldots = \beta_p = 0$
- The *alternative hypothesis* H_1 : is *at least one* of $\beta_1, \beta_2, \ldots, \beta_p$ *is not* equal to 0.

The F-test tells us whether there is at least one predictor that explains some of the outcome. It doesn't say which are the important ones. When there is a problem of *multicollinearity* (more later) it can be useful because the F-statistic may well be significant at the 5% level while all the predictors are not.

Q: Based on the output of `summary()` above, is the F-statistic significant at the 5% level? What does this imply for the model as a whole?

The F-test is a special case of the *partial F-test*. The partial F-test can be used to assess the joint significance of a group of predictors. Let us again consider a regression given by:

$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots \beta_p x_{pi}$

Let's say we want to add the following predictors: $x_{p+1}, \ldots, x_q$ where $q > p$ to the model. We may then run the following test:

- The *null hypothesis*, $H_0 :\ \beta_{p+1} = \beta_{p+2} = \ldots = \beta_q = 0$
- The *alternative hypothesis*, H_1 : is *at least one* of $\beta_{p+1}, \beta_{p+2}, \ldots, \beta_q$ *is not* equal to 0.

We are therefore asking whether the subset $x_{p+1}, \ldots, x_q$ of predictors are significant as a whole and adding them improves the regression model.

Adjusted $\mathbf{R}^2$

The Adjusted R^2 is designed to be interpreted in the same way as the R^2 but for multiple rather than simple linear regression. It takes into account the loss of information incurred by adding a non-significant variable. In practice if a variable is non-significant this can be deduced from the 95% confidence interval. There are two rules of thumb with using the Adjusted R^2 as a measure of model fit. For a good model:

1. It should be within 4% of the R^2 – otherwise there are probably too many non-significant predictors in the model
2. It should be close to 1 – same as the R^2.

Write down the formula for the Adjusted R^2:

`ifelse()`

`ifelse()` is a function that is necessary if using the `plot()` command to visualise the effect of adding a categorical variable to a regression. However it has other uses and is worth understanding. It is a one line function that works as follows:

`ifelse(A,IF A=TRUE DO THIS, ELSE (i.e. A=FALSE) DO THIS)`

There is also a longer version which has the `if` and the `else` separated and should be used when the commands for if and else are long and complex.

An easy example is of `ifelse()` is:

```
x<-1
ifelse(x>0,"hello","bye")
```

```
## [1] "hello"
```

```
x<- -1
ifelse(x>0,"hello","bye")
```

```
## [1] "bye"
```

The long version is

```
if(x>0)
{
  "hello"
} else {
  "bye"
}
```

```
## [1] "bye"
```

Lecture 5: Multiple Linear Regression, categorical variables

Learning Outcomes

Understanding Categorical variables

Using the partial F-statistic and `Anova()` in the "car" package to assess the significance of a categorical variable in linear regression

Understanding the concepts of overfitting and multicollinearity

Data

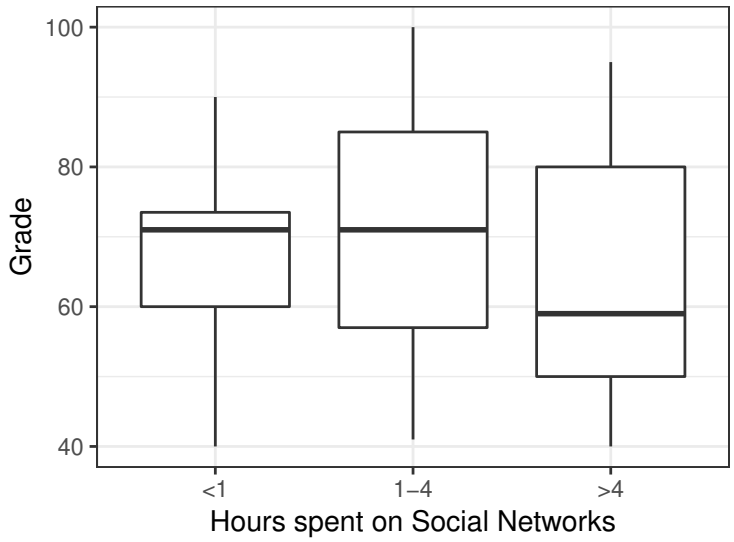
In this session we'll use the Study Skills data set. We'll initially include the categorical variable `Social.Net` and later the predictor `A.Level`. `A.Level` in principle has a range from 0-6, however in our data we only see the values 4,5 and 6 and only one person has a 4. As a consequence although it could be considered a continuous variable, we treat it as a categorical variable.

Name	Type	Description
Study Skills	continuous	0-56 Total study skills - higher is more skills
Grade	continuous	grades in stats exam out of 100
Social.Net	categorical	1 - less than 1 hour a week,2-between 1 and 4 hours,3-over 4 hours
A.Level	categorical	4,5,6 the grade received at A-level for mathematics or statistics.

```
Study_habits<-read.csv("Stats_study_habits_noout.csv")
```

`Social.Net`

`Social.Net` tells us how many hours students spend on social networks in a week. It is coded in the data as 1,2,3 representing less than one hour of social networking a week, between 1 and 4 hours and more than 4 hours a week respectively.



Q: Based on the boxplot do you think that "Social.Net" has an impact on Grade? Why?

Categorical variables

A variable is categorical when a variable has *discrete values* such that the *difference* between one value and another is not the same across variables. Ordinal variables are categorical variables with an inherent ordering (e.g. age or income bands)

Group Work

Q: Give an example of a variable with discrete values that is **not** categorical

Q: Give an example of a variable with discrete values that **is** categorical

Categorical variables behave the same way as binary variables except they have more levels. Consider adding `Social.Net` to the regression with `Study.Skills`. Like the binary variable `Interesting`, `Social.Net` results in multiple lines, one for each level of the variable.

Binary variables have 2 levels and are included as a single predictor in the regression. There is one line when the value of the binary variable =0 (or the *baseline* , e.g. "Female") and another one for when the binary variable=1 (or the other value, e.g. "Male"). For categorical variables with k levels there are k lines and they are included in the regression as **k-1** predictor variables. These predictors are binary and they are called *dummy* variables.

`Social.Net` has 3 levels coded as 1, 2, 3 corresponding to (<1,1-4,>4) indicating the number hours per week spent on social networks. So we define 2 new binary dummy variables:

$$\text{f.Social.Net14} = \begin{cases} 1 & \text{if Social.Net="1-4"} \\ 0, & \text{otherwise} \end{cases}$$

and

$$\text{f.Social.Net4} = \begin{cases} 1 & \text{if Social.Net=">4"} \\ 0, & \text{otherwise} \end{cases}$$

This means that when `f.Social.Net14=f.Social.Net4`=0 then`Social.Net`=<1. We then term `Social.Net`="<1" the *baseline*.

Dummy variables and baseline/reference levels: When we include a categorical variable with k levels in a regression we do two things:

1. First we choose a baseline/reference level. The baseline level of the categorical variable is the one that corresponds to 0 values for all the other levels. It is also that the others will be compared with. Often this is the most commonly occurring or the one of most interest.
2. Second we create k-1 binary dummies, one for each of the non-baseline levels. For a data point with level j (where j is between 1 and k-1), the dummies will all have 0 value except for the one corresponding to level j.

Formally in the model without interactions:

$$Grade = \beta_0 + \beta_1 Study.Skills + \beta_2 f.Social.Net14 + \beta_3 f.Social.Net4$$

This results in the following 3 lines:

`Social.Net`=1, ("<1"):

$$Grade = \beta_0 + \beta_1 Study.Skills$$

`Social.Net`=2, ("1-4")

$$Grade = (\beta_0 + \beta_2) + \beta_1 Study.Skills$$

`Social.Net`=3, (">4"):

$$Grade = (\beta_0 + \beta_3) + \beta_1 Study.Skills$$

We can see from the formulae above that the coefficient of `f.Social.Net14`, β_2 is the average difference in Grade between students who spend less than 1 hour on social networks and those who spend between 1 and 4 hours on social networks per week.

Q: What does the value of β_3 correspond to?

Q: How would you calculate the average difference in Grade for students spending between 1 and 4 and those spending more than 4 hours per week on social networks?

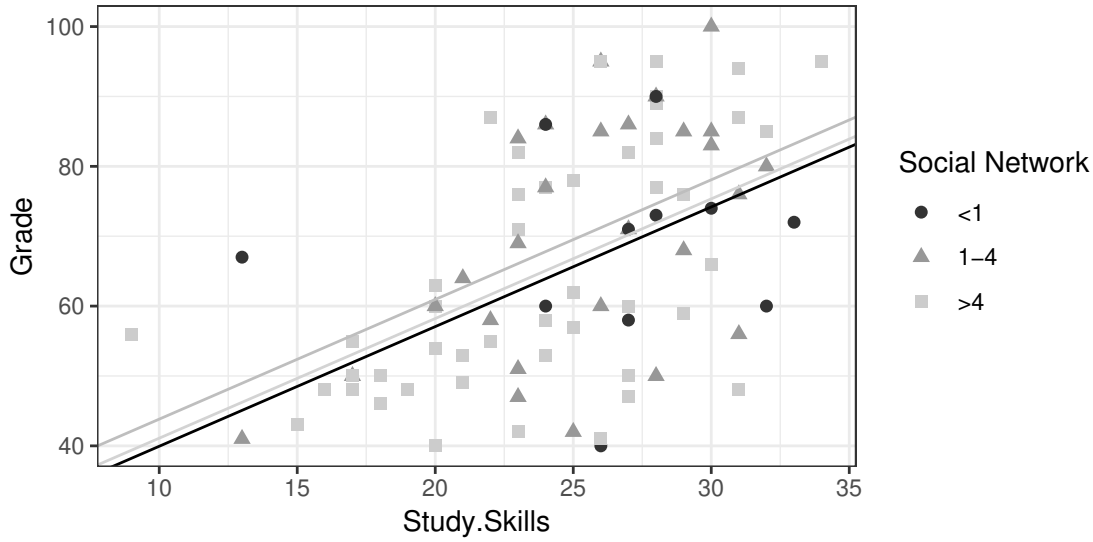
Let's run a regression. Notice that we have to tell `R` that `Social.Net` is a categorical variable by adding `as.factor()` or `factor()`.

Adding categorical predictors to regressions in `R`.

1. When you add a categorical predictor to `R` you do not need to add each of the dummy variables as `R` recognises that the predictor is categorical and automatically creates the dummies. Bear in mind that if you do not set the reference level then `R` will choose the first in alphabetical/numeric order as the reference level. I'll show you later how to set the reference level.
2. `factor()`: When a predictor is categorical and coded in the data set as an integer value then you must include it in a regression as `factor(x)` as otherwise `R` will not recognise it as a categorical variable and create dummies. It is not necessary when the variable is coded as text.

```
grade.lm.1<-lm(Grade~Study.Skills+factor(Social.Net),data=Study_habits)
display(grade.lm.1)
```

```
## lm(formula = Grade ~ Study.Skills + factor(Social.Net), data = Study_habits)
##                     coef.est coef.se
## (Intercept)         22.82     9.51
## Study.Skills         1.71     0.32
## factor(Social.Net)2  3.88     5.21
## factor(Social.Net)3  1.16     4.95
## ---
## n = 85, k = 4
## residual sd = 14.53, R-Squared = 0.28
```

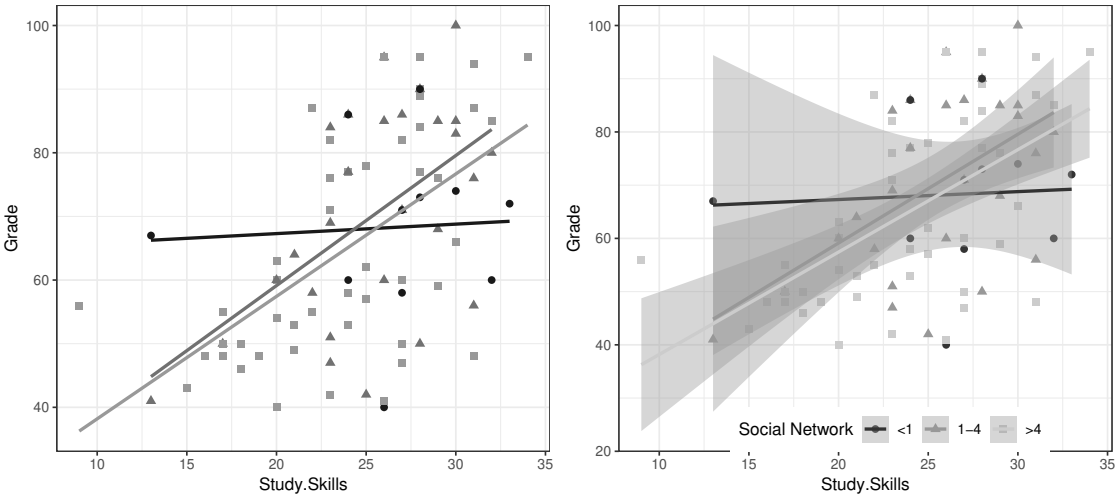


Q: Write down the regression lines for the 3 different levels of "Social.Net"

Note that the lines are very close to one another meaning that the difference between the baseline and the other categories is not large. This fits with the coefficients for the `Social.Net` categories not being significant at the 5% level.

Adding an interaction

If we add a `Study.Skills` and `Social.Net` interaction what does the regression look like?



The regression output is:

```
## lm(formula = Grade ~ Study.Skills * factor(Social.Net), data = Study_habits)
##                                   coef.est coef.se
## (Intercept)                        64.32    23.00
## Study.Skills                        0.15     0.85
## factor(Social.Net)2               -46.07    28.12
## factor(Social.Net)3               -45.37    24.99
## Study.Skills:factor(Social.Net)2    1.90     1.06
## Study.Skills:factor(Social.Net)3    1.78     0.94
## ---
## n = 85, k = 6
## residual sd = 14.36, R-Squared = 0.32
```

Q: Write down the regression lines for the 3 different levels of "Social.Net"

The interaction made a *big* difference! None of the predictors are significant any more! To see this, compare the output above to that of `grade.lm.1`.

There are two reasons.

1. The first is that by allowing the interaction, the line of best fit for those spending less than one hour on social networks per week (black dots in the scatterplot) is driven by the point on the far left.
2. The second reason is that the smaller sample size means that there is insufficient *power* to estimate the 6 parameters with more certainty – i.e. with confidence intervals that do not

include 0.

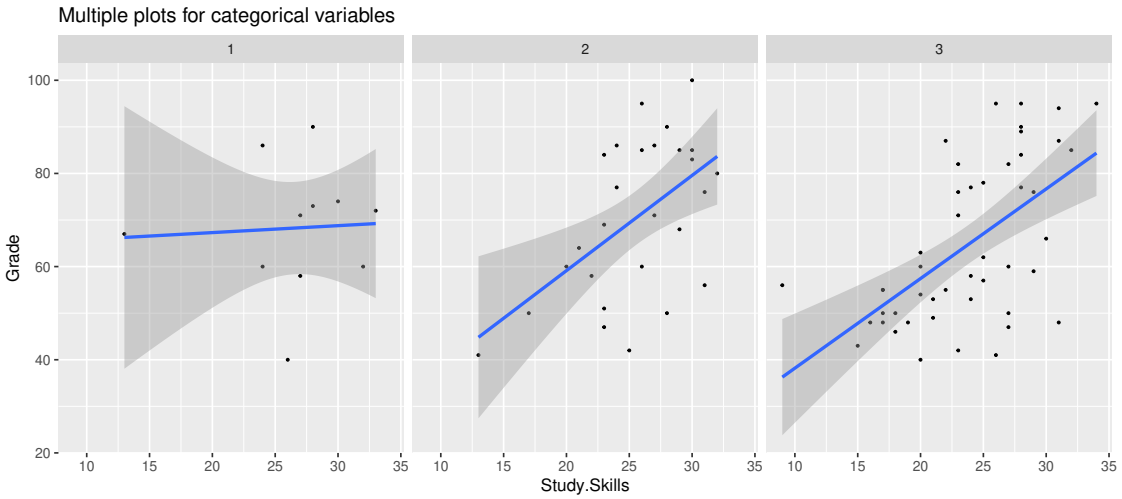
We can however do subset regression:

```
## # A tibble: 6 x 6
## # Groups:   Social.Net [3]
##   Social.Net term         estimate std.error statistic   p.value
##        <int> <chr>           <dbl>     <dbl>     <dbl>     <dbl>
## 1          1 (Intercept)    64.3      23.3       2.76  0.0220
## 2          1 Study.Skills    0.149     0.861     0.173 0.867
## 3          2 (Intercept)    18.2      16.5       1.11  0.279
## 4          2 Study.Skills    2.04      0.638     3.21  0.00367
## 5          3 (Intercept)    19.0       9.63      1.97  0.0553
## 6          3 Study.Skills    1.92      0.396     4.85  0.0000150
```

The table above shows in the firs column the value of `Social.Net` for this subset regression. So the first two rows correspond to the subset of students with `Social.Net=1`. The second column indicates whether the parameter is the intercept of the slope of `Study.Skills`. The remaining columns are the value of the estimate, its standard error, the value of the t-statistic and the corresponding p-value. We can also use the `facet_wrap()` function for `ggplot()`.

`facet_wrap()` is a convenient way of showing the effect of interactions by only looking at the points involved in each interaction.

```
p1 <- ggplot(data = Study_habits, aes(x = Study.Skills, y = Grade))
p1 <- p1 + geom_point(size = 0.5) + labs(title = "Multiple plots for categorical variables")
p1 <- p1 + facet_wrap(~factor(Social.Net)) + geom_smooth(method = "lm")
p1
```



Q: Based on the output of the subset regression and the plot, which of the slopes are significant at the 5

The partial F-statistic and ANOVA()

The *partial* F-statistic is useful when looking at the significance of categorical predictors in a regression as it asks whether a group of predictors are significant. Specifically, in order to understand whether a categorical variable as a whole is significant we run Analysis of Variance (ANOVA) which

essentially runs F-tests. This asks whether the means of the outcome is different in different levels of the categorical predictor. Contrast this to the t-tests for the coefficients of categorical dummy variables in linear regression which ask whether the mean outcome in each level is different from the mean outcome in the baseline level.

Assessing the overall significance of a categorical variable

Let's turn to our Study habits data and include `A.Level` (as a categorical variable) as well as `Social.Net` and `Interesting` which is binary.

```
## # A tibble: 3 x 3
##   A.Level Mean.Grade  Size
##     <int>      <dbl> <int>
## 1       4       42       1
## 2       5       67.1    22
## 3       6       67.2    62
```

The table above shows the value of `A.Level`, the mean grade in that level of `A.Level` and the number of students in that group. The regression test whether 67 is significantly different from 42. Even with a sample of 1 for `A.Level`=4 this comes out as just significant as shown below:

```
## lm(formula = Grade ~ Interesting + factor(Social.Net) + factor(A.Level),
##     data = Study_habits)
##                      coef.est coef.se t value Pr(>|t|)
## (Intercept)          23.44    16.44    1.43    0.16
## Interesting          15.86     3.63    4.37    0.00
## factor(Social.Net)2   2.70     5.49    0.49    0.62
## factor(Social.Net)3  -2.54     5.07   -0.50    0.62
## factor(A.Level)5     33.96    15.75    2.16    0.03
## factor(A.Level)6     32.92    15.42    2.13    0.04
## ---
## n = 85, k = 6
## residual sd = 15.10, R-Squared = 0.24
```

Rather than comparing whether levels of a categorical predictor as significantly different from one another in the regression a better approach is to try to understand whether the whole categorical variable is significant. We use the F-statistic for sub-groups of coefficients corresponding to the dummy variables associated with the categorical predictor. *Make sure to* capitalize *the A in* ***`Anova()`*** *or you'll get a different function.* You will also need to load and install the "car" package for this `Anova()` function.

```
Anova(anova.lm)
```

```
## Anova Table (Type II tests)
##
## Response: Grade
##                      Sum Sq Df F value    Pr(>F)
## Interesting          4361.6  1 19.1197 3.706e-05 ***
## factor(Social.Net)    449.8  2  0.9859    0.3777
## factor(A.Level)      1067.6  2  2.3400    0.1030
## Residuals           18021.7 79
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#car::Anova(anova.lm)
```

This tells us that as whole `A.Level` is not significant at the 5% level. I.e. the mean of `Grade` for all three levels of `A.Level` are not significantly different. This makes more sense as the means are

almost identical for the two levels with larger samples and not much can really be said about a sample of 1.

Overfitting

Overfitting happens when you use *too many predictors* in your linear model. In this case the model explains some of the random variation and therefore produces unreliable inference and predictions.

Ways to prevent this are:

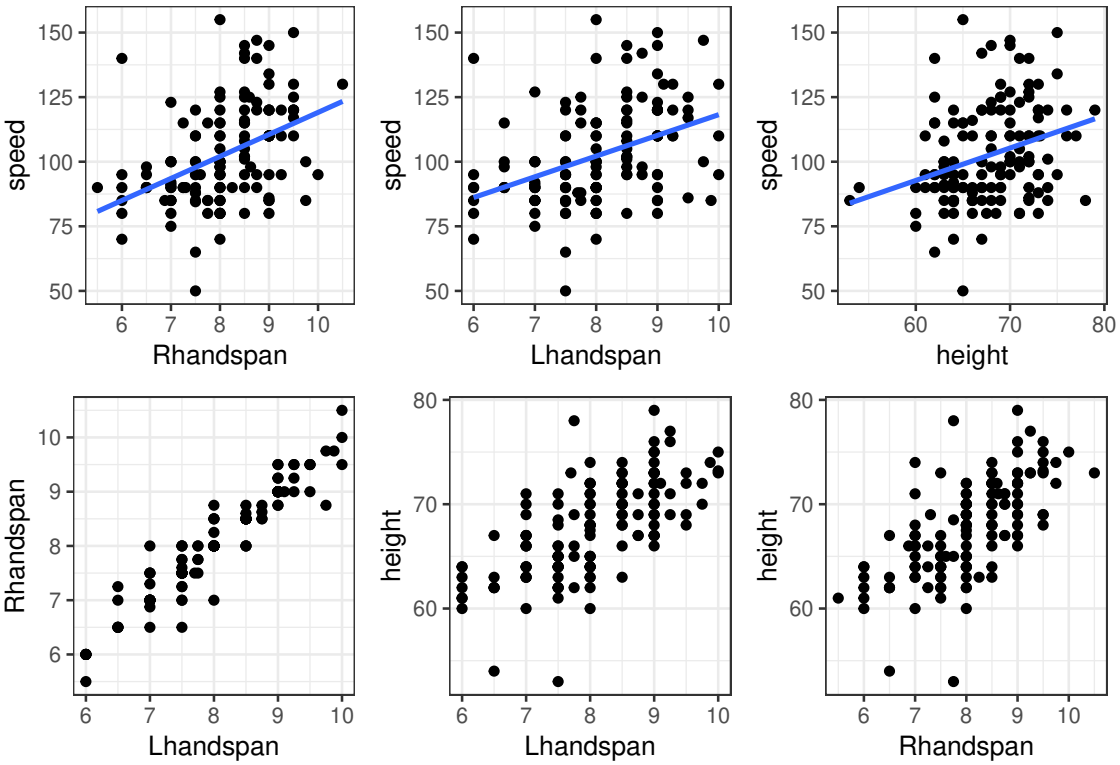
1. Avoid fitting too many non-significant predictors (choose non-significant predictors only if they are borderline non-significant or if they are important and you don't already have a lot of these)
2. If two models produce very similar predictions prefer the model with fewer predictors
3. Avoid fitting predictors that are correlated (more next)
4. Use more structurally complex models such as multi-level models (e.g. in ST308)

Multicollinearity (a.k.a collinearity)

We'll be looking at data which has the gender, height, handedness and right/left hand span of students and the fastest speed in mph (self reported) they have ever driven at. These are data from 1st year students in a US university.

```
Speed<-read.csv("Speed.csv",header=T)
head(Speed)
```

The outcome of interest in this case is `speed`. Let's see some plots:



Speed seems to have some linear relationships with at least `Rhandspan` and `Lhandspan`. However all three variables are very highly correlated amongst themselves (check `cor(height,Lhandspan)` as well).

What is the output of the regression with all continuous predictors?

```
## lm(formula = speed ~ height + Lhandspan + Rhandspan, data = Speed)
##             coef.est coef.se
## (Intercept) 28.34    22.01
## height       0.15     0.43
## Lhandspan   -1.76     5.14
## Rhandspan    9.74     5.11
## ---
## n = 139, k = 4
## residual sd = 17.34, R-Squared = 0.19
```

All three variables are non-significant at the 5% level. But from the plots they look as though at least one has a linear relationship with `speed`. Let's remove `Lhandspan` and re-run the regression:

```
## lm(formula = speed ~ height + Rhandspan, data = Speed)
##             coef.est coef.se
## (Intercept) 28.58    21.93
## height       0.13     0.43
## Rhandspan    8.13     2.02
## ---
## n = 139, k = 3
## residual sd = 17.29, R-Squared = 0.19
```

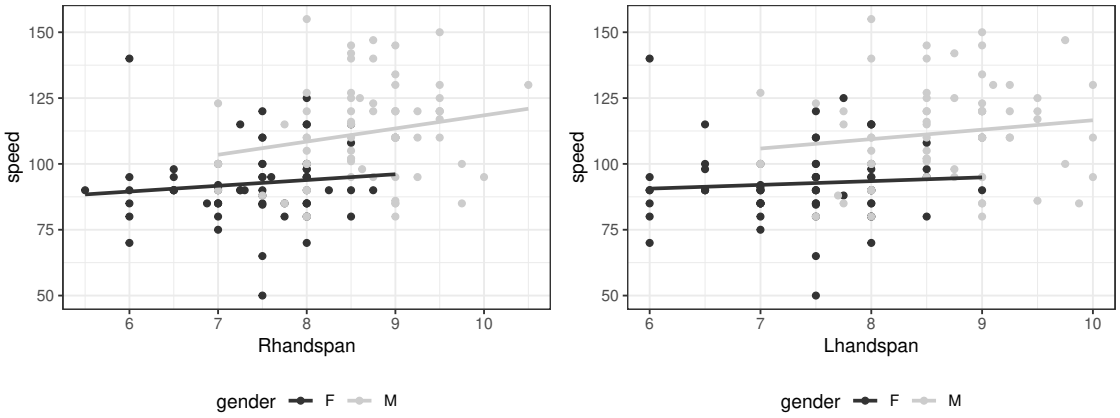
Now `Rhandspan` *is* significant at the 5% level.

What happens is that the `Lhandspan` and `Rhandspan` are so highly correlated (i.e. highly *dependent*) that they are almost the same variable. This means that the X matrix of values of the predictors is not full rank – the rank of a matrix is the maximal number of linearly independent columns of X. This implies that X^TX cannot be inverted. Because the vector of regression coefficients $\hat{\beta}$ is a linear transformation of the inverse of X^TX ($\hat{\beta} = (X^TX)^{-1}X^Ty$) we cannot estimate it.

Out of curiosity what happens if we add `gender` to the regression?

```
## lm(formula = speed ~ height + Rhandspan + gender, data = Speed)
##             coef.est coef.se
## (Intercept) 120.87    28.58
## height       -1.03     0.47
## Rhandspan     5.09     2.00
## genderM      19.69     4.26
## ---
## n = 139, k = 4
## residual sd = 16.12, R-Squared = 0.30
```

The reason why is most clearly seen in these plots:



In practice multicollinearity is really only a problem when variables are *very* highly correlated. A statistic commonly used to diagnose multicollinearity is the Variance Inflation Factor (VIF). When this is more than 10 there is a problem.

```
vif(Speed.lm.1)
```

```
##    height Lhandspan Rhandspan
##  1.812906 11.304126 11.243165
```

As we can see there is a definite problem with `LhandSpan` and `Rhandspan`.

Some things to note about **collinearity**

1. Collinearity statistics are based on correlations that are *linear* rather than non-linear. You may well have predictors that are highly correlated to one another (dependent) but in a non-linear way.
2. Collinearity refers to the dependence/correlation between two or more *predictors* and not to how they *interact* in the outcome. Correlation between predictors does not depend on the outcome; `Rhandspan` and `Lhandspan` are correlated regardless of whether we consider speed or some other outcome. However two predictors may interact (be dependent) for one outcome and not for another. More formally collinearity happens when predictors are *marginally* dependent whereas interactions happen the predictors are dependent *conditional* on the outcome (but possibly marginally *independent*).

Exercises on how to interpret/visualise interactions between categorical variables are given at the back of the book

Code for the plots in this Lecture:

```
## ----setup, include=FALSE-----------------------------------------------
library(ggplot2)
library(gridExtra)

## ------Data--------------------------------------------------------------
Study_habits <- read.csv("Stats_study_habits_noout.csv")

## ----Boxplot ----------------
p1 <- ggplot(Study_habits, aes(x = factor(Social.Net), y = Grade, fill = factor(Social.Net)))
p1 <- p1 + geom_boxplot(fill = "white")
p1 <- p1 + xlab("Hours spent on Social Networks")
p1 <- p1 + scale_colour_grey(start = 0.1, end = 0.6) + theme_bw()

p1 <- p1 + scale_x_discrete(labels = c(`1` = "<1", `2` = "1-4", `3` = ">4"))
p1
```

```
## ------Regression-------------------------------------------------------------
grade.lm.1 <- lm(Grade ~ Study.Skills + factor(Social.Net), data = Study_habits)
display(grade.lm.1)

## ----Plot without interactions-----------
p1 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, shape = factor(Social.Net),
    colour = factor(Social.Net)))
p1 <- p1 + geom_point(aes(shape = factor(Social.Net)), size = 2)
p1 <- p1 + geom_abline(intercept = coef(grade.lm.1)[1], slope = coef(grade.lm.1)[2],
    color = "black")
p1 <- p1 + geom_abline(intercept = (coef(grade.lm.1)[1] + coef(grade.lm.1)[3]),
    slope = coef(grade.lm.1)[2], color = "gray")
p1 <- p1 + geom_abline(intercept = (coef(grade.lm.1)[1] + coef(grade.lm.1)[4]),
    slope = coef(grade.lm.1)[2], color = "lightgray")
p1 <- p1 + theme_bw()
p1 <- p1 + scale_colour_grey(name = "Social Network", breaks = c("1", "2", "3"),
    labels = c("<1", "1-4", ">4"))
p1 <- p1 + scale_shape_discrete(name = "Social Network", breaks = c("1", "2",
    "3"), labels = c("<1", "1-4", ">4"))
# p1<-p1 +labs(color='Social Network',shape='Social Network')
p1
```

```
## ----Plots with interactions and with/without confidence
## bands------------------------------
p1 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, shape = factor(Social.Net),
    color = factor(Social.Net)))
p1 <- p1 + geom_point(size = 2) + theme_bw()
p1 <- p1 + scale_colour_grey(start = 0.1, end = 0.6)
p1 <- p1 + geom_smooth(method = "lm", se = FALSE)
p1 <- p1 + theme(legend.position = "none")

p2 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, shape = factor(Social.Net),
    color = factor(Social.Net)))
p2 <- p2 + geom_point(size = 2) + theme_bw()
p2 <- p2 + geom_smooth(method = "lm")
p2 <- p2 + theme(legend.position = c(0.55, 0.05), legend.direction = "horizontal")
p2 <- p2 + scale_colour_grey(name = "Social Network", breaks = c("1", "2", "3"),
    labels = c("<1", "1-4", ">4"))
```

```
p2 <- p2 + scale_shape_discrete(name = "Social Network", breaks = c("1", "2",
    "3"), labels = c("<1", "1-4", ">4"))
# grid.arrange allows us to put the figures together in a single plot
grid.arrange(p1, p2, nrow = 1)
```

```
## ----Plots using facet_wrap-----------------------------------------------
p1 <- ggplot(data = Study_habits, aes(x = Study.Skills, y = Grade))
p1 <- p1 + geom_point(size = 0.5) + labs(title = "Multiple plots for categorical variables")
p1 <- p1 + facet_wrap(~factor(Social.Net)) + geom_smooth(method = "lm")
p1
```

```
## ----Initiall scatterplots for collinearity
## question-----------------------------------
p1 <- ggplot(Speed, aes(x = Rhandspan, y = speed)) + geom_point() + theme_bw() +
    geom_smooth(method = "lm", se = FALSE) + scale_color_grey()
p2 <- ggplot(Speed, aes(x = Lhandspan, y = speed)) + geom_point() + theme_bw() +
    geom_smooth(method = "lm", se = FALSE) + scale_color_grey()
p3 <- ggplot(Speed, aes(x = height, y = speed)) + geom_point() + theme_bw() +
    geom_smooth(method = "lm", se = FALSE) + scale_color_grey()
p4 <- ggplot(Speed, aes(x = Lhandspan, y = Rhandspan)) + geom_point() + theme_bw() +
    scale_color_grey()
p5 <- ggplot(Speed, aes(x = Lhandspan, y = height)) + geom_point() + theme_bw() +
    scale_color_grey()
p6 <- ggplot(Speed, aes(x = Rhandspan, y = height)) + geom_point() + theme_bw() +
    scale_color_grey()
# grid.arrange allows us to put the figures together in a single plot
grid.arrange(p1, p2, p3, p4, p5, p6, nrow = 2)
```

```
## -------------Plot to show the effect of gender on the
## regressions--------------
p1 <- ggplot(Speed, aes(x = Rhandspan, y = speed, colour = gender)) + geom_point() +
    theme_bw() + scale_color_grey() + theme(legend.position = "bottom") + geom_smooth(method
    se = FALSE)
p2 <- ggplot(Speed, aes(x = Lhandspan, y = speed, colour = gender)) + geom_point() +
    theme_bw() + scale_color_grey() + theme(legend.position = "bottom") + geom_smooth(method
    se = FALSE)
# grid.arrange allows us to put the figures together in a single plot
grid.arrange(p1, p2, nrow = 1)
```

Workshop 5: Multiple Linear regression: Outliers

Learning outcomes

How to identify and deal with outliers

Outliers

Outliers are defined in many ways.

- Points that are "away from the main trend of the data". We can think in terms of x-direction, y-direction or both.
- Points that affect the estimates of the coefficients and standard errors and/or well as the fit of the model to the data.
- Points that have large residuals

Some points fulfill only one of the criteria above whereas others fulfill two or three.

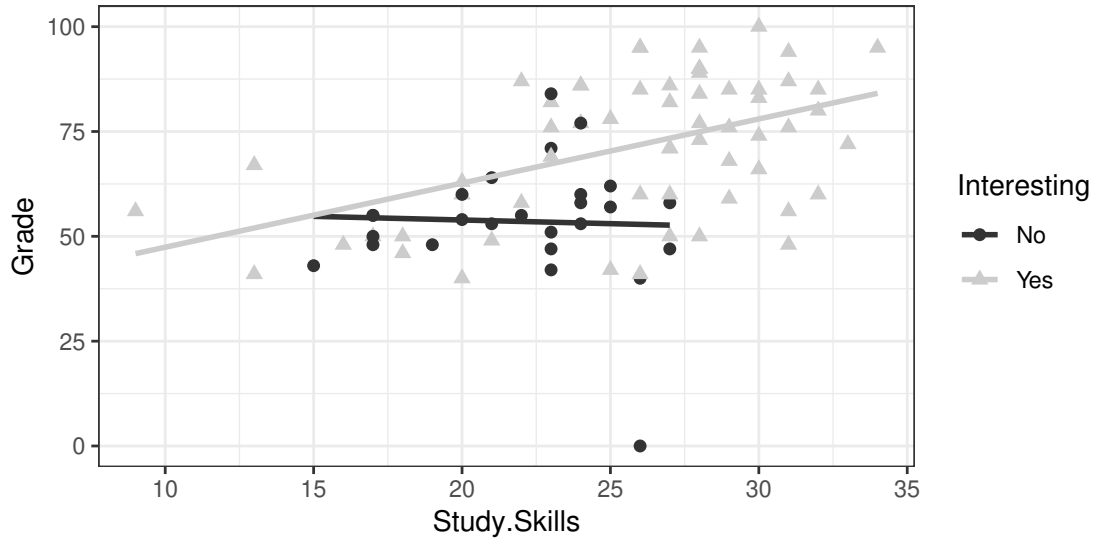
Let's load the Study habits data set and run the model with the interaction:

```
Study_habits<-read.csv("Stats_study_habits.csv")
grade.lm.2<-lm(Grade~Study.Skills*factor(Interesting),data=Study_habits)
display(grade.lm.2)
```

```
## lm(formula = Grade ~ Study.Skills * factor(Interesting), data = Study_habits)
##                                   coef.est coef.se
## (Intercept)                        57.39    19.69
## Study.Skills                       -0.17     0.88
## factor(Interesting)1              -25.29    21.97
## Study.Skills:factor(Interesting)1   1.70     0.96
## ---
## n = 86, k = 4
## residual sd = 15.02, R-Squared = 0.35
```

> **Why is everything non-significant?**
> The sample sizes for the not-interested group is 27<30. Also, the model has to estimate 5 parameters. This means that the model does not have sufficient power and coefficients appear less significant than in a model without the interactions.

```
p1<-ggplot(Study_habits, aes(x=Study.Skills, y=Grade,  color=factor(Interesting)))
p1<- p1 + geom_point(size=2)
p1<-p1+geom_smooth(method="lm", se=FALSE)
#legend
p1<-p1+scale_colour_discrete(name  ="Interesting",
                             breaks=c("0", "1"),
                             labels=c("No", "Yes"))
p1
```



Note:The plot in your notes is different from the one on your screen as it is optimised for printing and includes shapes as well as colours for the points in order to make it easier to distinguish between them.

Looking carefully we see that there are a few points that stand out as being far away from the main *group* of the points. The point with `Grade` above 60 and `Study.Skills` below 15. The point with `Grade` around 55 and `Study.Skills` below 10 and the point with 0 `Grade`. This is somewhat arbitrary, we could also have included the point with `Grade` over 80 and `Interesting`=0 (No).

Let's pick these out and take a closer look:

```
out.1<-which(Study_habits$Grade==min(Study_habits$Grade))
out.2<-with(Study_habits, which(Grade<60 & Study.Skills<10))
out.3<-with(Study_habits, which(Grade>60 & Study.Skills<15))
Study_habits[c(out.1,out.2,out.3),c(2,7,8)]
```

```
##    Interesting Study.Skills Grade
## 1            0           26     0
## 14           1            9    56
## 69           1           13    67
```

Outlier questions

For any potential outlier, we need to ask ourselves the following questions:

1. Is it a feasible result – i.e. is it the result of a typo/error?
    a. If it *is not* a feasible result then it is often fine to remove the point, but this needs to be argued and justified.
    b. If it *is* a feasible result then should we investigate what effect it is having on the regression.
2. If it is a feasible point and does have an effect on the data?
    a. If the point is feasible but is extreme in many ways, or "odd", "atypical") then we can present multiple regressions and point out that for prediction a model without the point may be better.
    b. If the point is one of many others picked up by the outlier statistics below then unless we have strong reasons to believe that the points should be analysed separately or removed for other reasons, we can simply display the statistics and move on. **Do not use the statistics as a reason to remove points to improve the regression diagnostics.**

Q: Looking at the point `out.1`, ask yourself the outlier questions. How might you deal with this point?

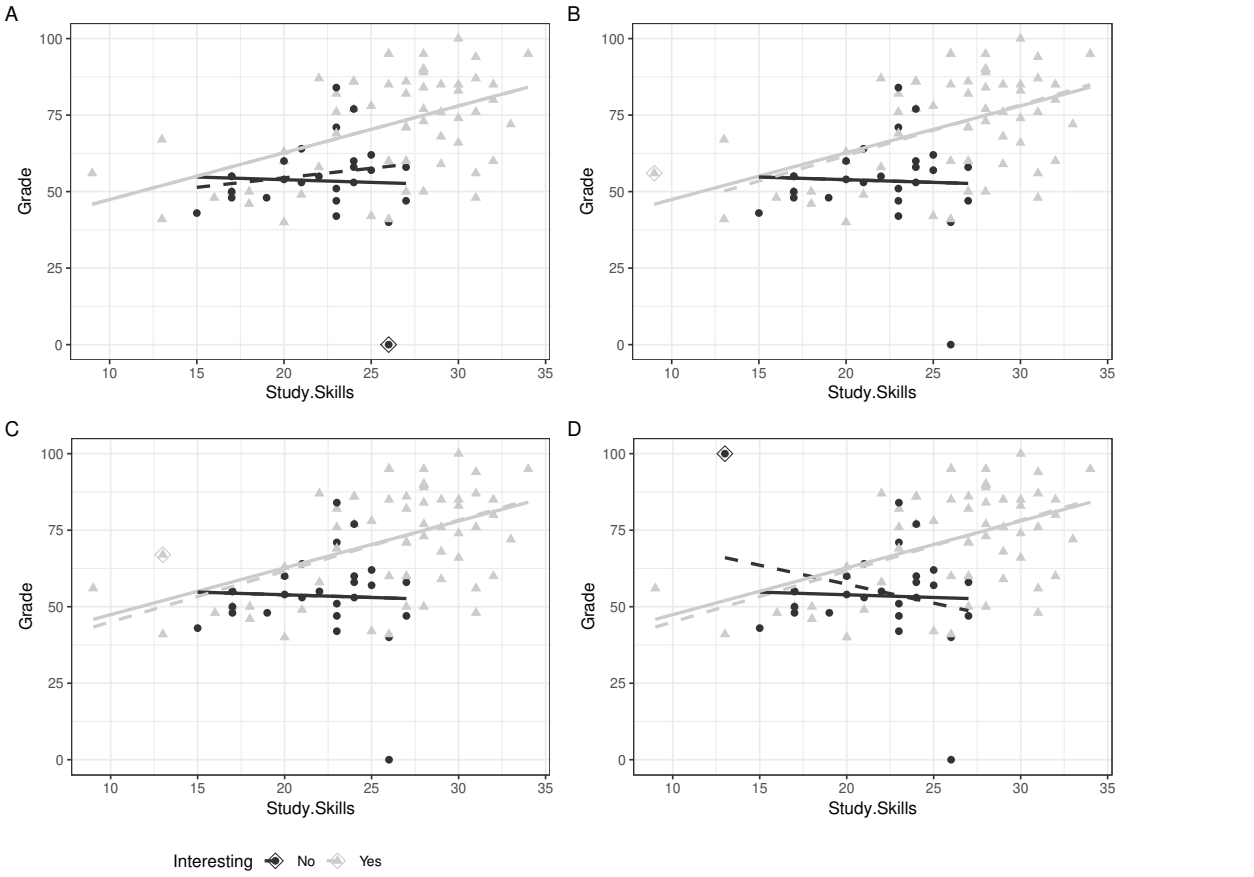
If it is a typo we can remove the point and re-run the regression:

```
Study_habits.out.1<-Study_habits[-out.1,]
grade.lm.no.out<-lm(Grade~Study.Skills*factor(Interesting),data=Study_habits.out.1)
display(grade.lm.no.out)
```

```
## lm(formula = Grade ~ Study.Skills * factor(Interesting), data = Study_habits.out.1)
##                                  coef.est coef.se
## (Intercept)                      42.03    18.49
## Study.Skills                      0.62     0.84
## factor(Interesting)1             -9.93    20.54
## Study.Skills:factor(Interesting)1  0.91     0.90
## ---
## n = 85, k = 4
## residual sd = 13.79, R-Squared = 0.35
```

The coefficients appear quite different, but what is actually happening? Look at plot A in your notes. Code to reproduce it is below.

```
p1 <- ggplot(Study_habits, aes(x = Study.Skills, y = Grade, color = factor(Interesting)))
p1 <- p1 + geom_point(size = 2)
p1 <- p1 + geom_point(data = Study_habits[out.1, ], color = "black")
p1 <- p1 + geom_smooth(method = "lm", se = FALSE)
p1 <- p1 + scale_colour_discrete(name = "Interesting", breaks = c("0", "1"),
    labels = c("No", "Yes"))
p2 <- ggplot() + geom_smooth(data = Study_habits.out.1, aes(x = Study.Skills,
    y = Grade, color = factor(Interesting)), method = "lm", se = FALSE, linetype = "dashed")
```



In the plots A-D above the regression lines for the data with all the points are solid and those with

a point missing are dashed. The point that has been removed is surrounded by a small diamond.

Q: For plot A where we have removed "out.1" what has changed and what has stayed the same? Does this make sense?

This point appears to be somewhat *influential* in the model with an interaction (although everything remains non-significant). By *influential* we mean is that it changes regression coefficients.

The other potentially outlying points are most likely not mistakes so we cannot remove them. However we can investigate whether it is *influential* by removing it from the regression and investigating whether it changes the coefficients and looking at plot B.

```
## lm(formula = Grade ~ Study.Skills * factor(Interesting), data = Study_habits.out.2)
##                                    coef.est coef.se
## (Intercept)                         57.39    19.74
## Study.Skills                        -0.17     0.89
## factor(Interesting)1               -28.78    22.52
## Study.Skills:factor(Interesting)1    1.83     0.98
## ---
## n = 85, k = 4
## residual sd = 15.06, R-Squared = 0.35
```

Removing point `out.2` in plot B only just changes the slope for the Interested. This is because it is in the "middle" of the range of `Study.Skills` and is therefore not very influential. Finally let us consider the third point `out.3` which is removed in plot C. As with the previous point they cannot be removed on the basis that it is incorrect, similarly it does not appear to be influential.

Modify a point

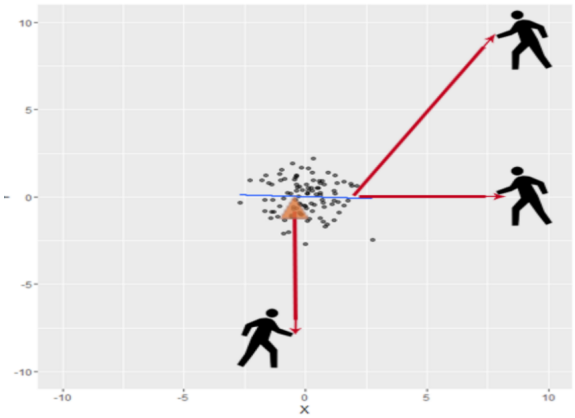
Let's modify `out.3` so that it becomes influential. We replace it's value of `Grade` with 100 and Interesting with 0 instead of 1. The corresponding plot is D.

```
Study_habits.newout<-Study_habits
Study_habits.newout[out.3,]$Grade<-100
Study_habits.newout[out.3,]$Interesting<-0
```

Modifying `out.3` made a big difference! It has significantly changed the slope of the regression for the un-interested.

Q: Using plots C and D and any other information, explain why the the new "out.3" has a larger impact on the regression line that the old "out.3"

A visual explanation of how outliers affect a regression line.



Outlier statistics

Let's use the data with the modified `out.3` to examine the outlier statistics. This is for *exploratory* analysis. DO NOT EVER IN A PROJECT USE THESE STATISTICS TO REMOVE A BUNCH OF POINTS WITHOUT A CAREFUL AND DETAILED JUSTIFICATION!

- *Standardised residuals* : These are residuals that have been transformed so they approximately follow a standard normal distribution. If they are above 3 in terms of absolute value then investigate
- The *leverage values* : These are like residuals but in the *horizontal* direction. The rule of thumb is that a leverage value greater than $2\frac{p}{n}$ is a potential problem. p is the number of predictors+1 (i.e. the number of estimated parameters) and n is the sample size.
- *Cook's distance*: $D_i = \frac{1}{p}\frac{h_i}{1-h_i}\tau_i^2$ where τ_i is the studentised residual[1]. If these are above 1 then investigate
- *DIFFTS* : An alternative to the Cook's distance. If this is above 1 for small data sets (<30) or if it is above $2\sqrt{\frac{p}{n}}$ for large data sets then investigate.

First, let's run the regression:

```
grade.out<-lm(Grade~Study.Skills*Interesting,data=Study_habits.newout)
display(grade.out)
```

```
## lm(formula = Grade ~ Study.Skills * Interesting, data = Study_habits.newout)
##                           coef.est coef.se
## (Intercept)                82.12    17.82
## Study.Skills               -1.24     0.81
## Interesting               -53.71    20.79
## Study.Skills:Interesting    2.90     0.90
## ---
## n = 86, k = 4
## residual sd = 15.52, R-Squared = 0.33
```

In our case p=4, n=86

Now let's set up a function that will identify the outliers automatically when we pass it a regression. This is in Moodle.

[1] A studentised residual for a point i rather than a standardised residual is when you fit a regression without i (say lm_{-i}) and then calculate the difference between y_i and the estimated/predicted $\hat{y}_i$ using lm_{-i}. You then get a studentised residual for each point i. The studentised residual will typically be larger than the standardised residual for outlying points.

```r
# This function displays the outlier statistics. Note that there is no limit
# to the number of these so it could blow up if there are a lot of them. You
# can modify the function to deal with this if you like

show_outliers <- function(the.linear.model, topN) {
    # length of data
    n = length(fitted(the.linear.model))
    # number of parameters estimated
    p = length(coef(the.linear.model))
    # standardised residuals over 3
    res.out <- which(abs(rstandard(the.linear.model)) > 3)  #sometimes >2
    # topN values
    res.top <- head(rev(sort(abs(rstandard(the.linear.model)))), topN)
    # high leverage values
    lev.out <- which(lm.influence(the.linear.model)$hat > 2 * p/n)
    # topN values
    lev.top <- head(rev(sort(lm.influence(the.linear.model)$hat)), topN)
    # high diffits
    dffits.out <- which(dffits(the.linear.model) > 2 * sqrt(p/n))
    # topN values
    dffits.top <- head(rev(sort(dffits(the.linear.model))), topN)
    # Cook's over 1
    cooks.out <- which(cooks.distance(the.linear.model) > 1)
    # topN cooks
    cooks.top <- head(rev(sort(cooks.distance(the.linear.model))), topN)
    # Create a list with the statistics -- cant do a data frame as different
    # lengths
    list.of.stats <- list(Std.res = res.out, Std.res.top = res.top, Leverage = lev.out,
        Leverage.top = lev.top, DFFITS = dffits.out, DFFITS.top = dffits.top,
        Cooks = cooks.out, Cooks.top = cooks.top)
    # return the statistics
    list.of.stats
}
```

Now let's run it with our regression and consider the top 5 values for each statistic. For each statistic the outlier that fulfills the rule is shown, so for Std.Res it is the first row. Then the top highest values are shown for each criterion. Again there are no values that fulfill the Cook's distance > 1.

```r
#Apply to grade.out and look at the top 5 values in each statistic
grade.out.stats<-show_outliers(grade.out, 5)
grade.out.stats
```

```
## $Std.res
## 1
## 1
##
## $Std.res.top
##        1       69       52       32       44
## 3.367575 2.515365 2.092206 1.996802 1.989210
##
## $Leverage
##  3  7 14 45 49 51 56 57 62 69
##  3  7 14 45 49 51 56 57 62 69
##
## $Leverage.top
##        69        14         3        45        62
## 0.2430989 0.2130087 0.1592787 0.1316701 0.1133387
```

```
##
## $DFFITS
## 14 69
## 14 69
##
## $DFFITS.top
##        69        14        32        76        80
## 1.4748412 0.4768189 0.4239463 0.3822975 0.2453731
##
## $Cooks
## named integer(0)
##
## $Cooks.top
##         69          1          3         14         32
## 0.50802617 0.27099679 0.09877868 0.05695020 0.04327553
```

In practice you often get points that violate more than one criteria. Look for these points. We use the function `intersect()` which gives us the intersection of its two arguments.

```
Reduce(intersect,list(grade.out.stats$DFFITS,grade.out.stats$Leverage))
```

```
## [1] 14 69
```

In our case there are no outliers for standardised residuals or Cook's distance but you could put the top 3 of the Cook's distance into the `intersect()` function. The points that are common between the DFFITS and the leverage are in fact the point we modified (69) and the other low `Study.Skills` high `Grade` point. (14). The `Grade`=0 point was not identified as a problem in the intersection (although it does appear in some of the top 5 lists – it corresponds to row 1.)

How to deal with outliers:

1. You should try to identify potential outliers (or groups of outliers) from the beginning (e.g. from scatterplots or otherwise)
2. However, you should deal with them towards the end of the analysis (after transformations typically – we'll see what transformations are after Reading Week)
3. If you can argue that an outlying point is a mistake (typo or similar) then you can remove it from your analysis – although you should report that you've done that
4. If you cannot argue that an outlying point is a mistake then:
   i. investigate what makes it an outlier;
   ii. does removing them change any of the regression parameters?;
   iii. if not then keep them in the regression and stop worrying about them;
   iv. if they do change any regression parameters present a regression with and without the point(s) and discuss which regression you prefer for understanding/prediction
   v. if there are multiple outliers investigate whether they have common values for (some of) their predictors (e.g. all white men under 15);
   vi. if they have common factors and there are a sufficient number of them consider running a subset regression for these points;
   vii. if there are not enough of them or they have no common factors go back to point iii and iv.

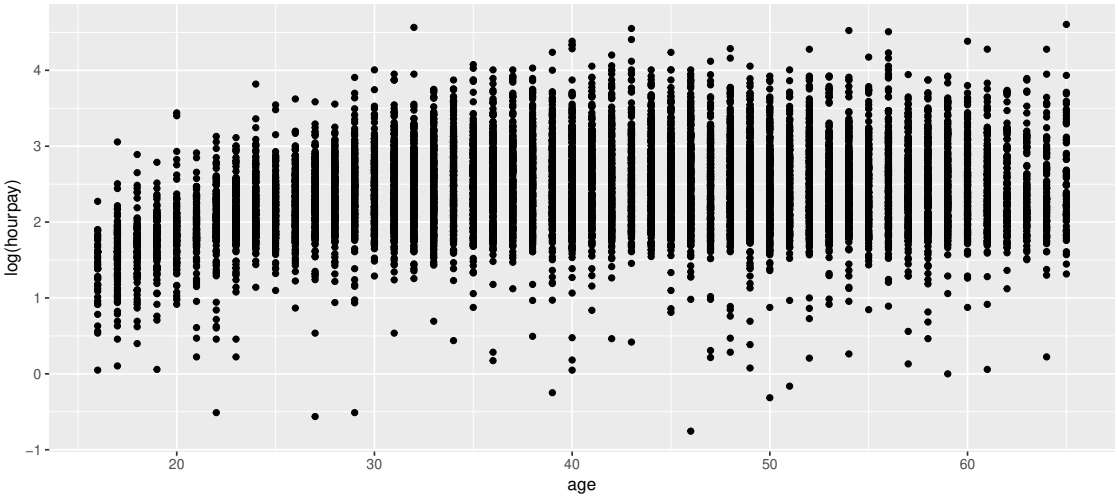
Using a Large data set

When using a large data set the statistics above can often identify very many outliers. Let's use the Hourly pay dataset:

```
Hourly_Pay <- read.csv("Large_Hourly_Pay.csv", header = TRUE, row.names = NULL)
HP.lm <- lm(log(hourpay) ~ age + factor(gender) + HighestQual, data = Hourly_Pay)
display(HP.lm)
```

For these data, I haven't printed output in the book as it is rather a lot. The code is available on Moodle. Below is the code for a plot of the `log(hourpay)` against age using shape to distinguish men from women and colour for `HighestQual`. We can see that there are a lot of high residuals.

```
ggplot(Hourly_Pay, aes(x = age, y = log(hourpay))) + geom_point()
```



Let's run our outlier function but only look at the top 10 values:

```
HP.out<-show_outliers(HP.lm,10)
```

```
## [1] "Standardised residuals"
```

```
##     7125     3953     7714     2908     7715     7126     7127     7123
## 6.438407 5.760791 5.731614 5.587819 5.417449 5.384487 5.273021 5.232181
##      757     7124
## 5.196158 5.093280
```

```
## [1] "Leverage values"
```

```
##        1045         280         751         403         270        5906
## 0.002513128 0.002513128 0.002485056 0.002485056 0.002485056 0.002472609
##        2903         705        7071        3074
## 0.002472609 0.002445244 0.002433410 0.002433410
```

```
## [1] "DFFITS"
```

```
##      9175      4643        23      3702      3955      3956      8554
## 0.1700181 0.1526804 0.1410194 0.1365560 0.1339173 0.1327499 0.1322160
##      8078      3234      3241
## 0.1233356 0.1213785 0.1210240
```

```
## [1] "Cook's d"
```

```
##        2908        7127       10235        1253        7122        9175
## 0.008333627 0.005859104 0.005294864 0.005192339 0.004657007 0.004123694
##        7126        4643        2945        5909
## 0.003417683 0.003326546 0.003239666 0.003172481
```

There are a lot of outliers! It is worth noting that the Cook's distances are all way below 1. In this sort of situation it is difficult to justify removing any of the outliers. It may be worth looking at these values and trying to identify if they have common elements:

```
common.out<-intersect(intersect(HP.out$Std.res,HP.out$DFFITS),HP.out$Leverage)
common.out
```

```
## [1]   23 3234 3702 3956 4643 9175
```

```
#options makes sure the output fits on the page
options(width=100)
Hourly_Pay[common.out,]
```

```
##      OwnHouse gender age AgeYoungDep Ethnicity hourpay TotalHoursWorked HighestQual
## 23        Own      0  55          19     White   65.00                4        None
## 3234      Own      0  65          19     White   51.09               35        None
## 3702      Own      1  55          19     White   34.72               36       Other
## 3956     Rent      0  26           3     Other   37.43               37        None
## 4643      Own      0  52           8     White   50.61               38       Other
## 9175      Own      1  46          19     White   46.16               50       Other
```

```
summary(Hourly_Pay)
```

Q: Compare the values of the predictors for these common outliers to the mean/median values given in `summary()`. Can you identify why they are outliers or influential points? Should you remove them?

Added variable plots

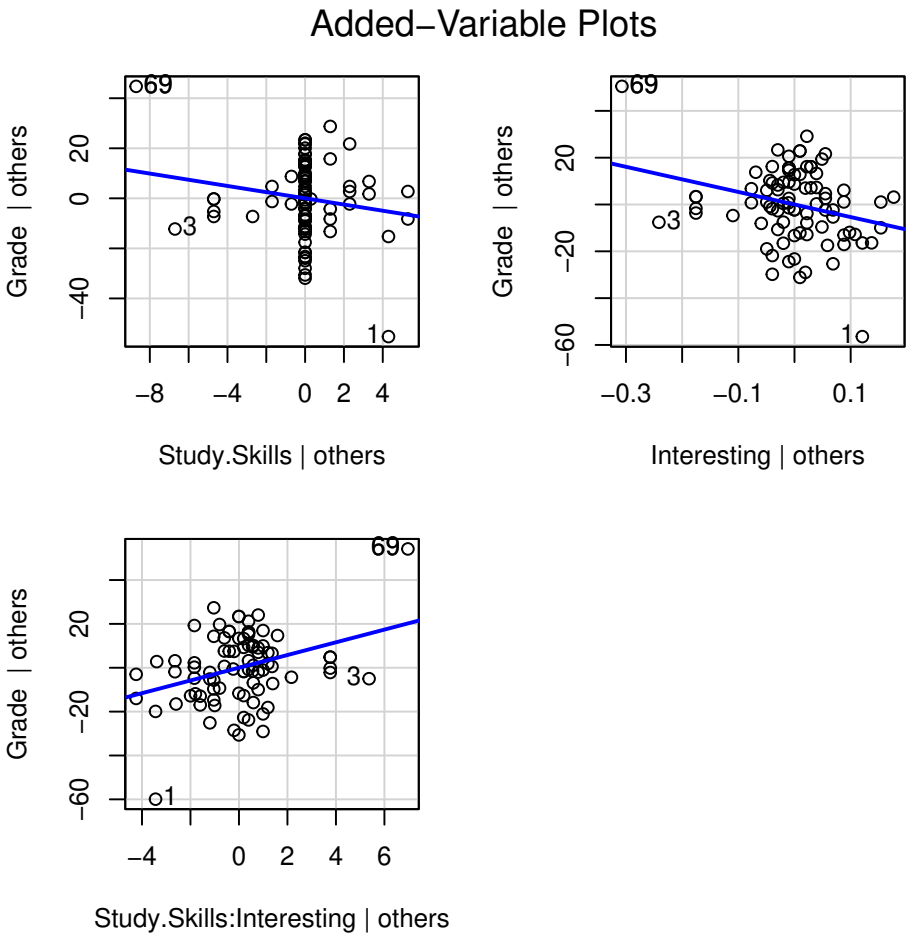
We can also use the *added variable plots* more commonly known as partial regressions to assess potential outliers. These plots are also useful to see whether a variable is a good predictor taking into account other variables in a multiple regression model. If we are interested in the relative importance of e.g. "x_1" in the regression $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$ then:

1. Run $y = \beta_0 + \beta_2 x_2 + \beta_3 x_3$ *without* x_1 and take the residuals (r_1)
2. Run $x_1 = \gamma_0 + \gamma_1 x_2 + \gamma_2 x_3$: i.e. how x_1 depends on the other predictors, and take these residuals (r_2)
3. Plot r_1 vs r_2

The `avPlots()` function in the `car` package identifies 4 points: the two with the highest residual and the most extreme horizontal values (called partial leverage which we get from

```
sort(lm.influence(grade.lm.int)$hat)
```

```
avPlots(grade.out)
```



The added variable plots identify the point we modified (69) and the 0 `Grade` point. We can also see that the slopes of all three plots are not horizontal, indicating that all three main effects are probably significant.

Lecture 6: Transformations

Learning outcomes

Understand what transformations do and when they are appropriate (for outcomes and/or predictors)

Understand how to interpret the regression coefficients for the transformed models

Why transformations?

Often the data in their raw form cannot appropriately be modelled using a linear regression: Some examples are when:

- It is harder to interpret the output with the data in their raw form (e.g. the intercept)
- The relationship between one or more of the predictors and the outcome is non-linear.
- The residual assumptions are violated (in particular there is a funnel shape in the residual vs fitted plot)
- There are clustered outliers (sometimes a transform can help)
- There is a theoretical reason why a transformed version of the outcome should be analysed.

Both outcome and predictors can be transformed.

Centering the predictors

We saw in the data on Study habits that when we include Study Skills as a predictor the intercept refers to a situation where the Study Skills are 0 which is something that does not occur in the data (and is unlikely to ever be the case, a student who has made it into university must have some study skills). Similarly when we look at the pay data it is meaningless to consider the hourly pay of a person who is aged 0 years (or even up to 16 years of age).

We look at data from the British Panel Household survey from the early 90s. Note that in these data there were no options for gender other than male and female. Nowadays the Understanding Society survey accommodates other expressions of sexual identity. In these data females are `gender`=1 but we create a new predictor called `genderCat` where females and males are represented by `F` and males by `M` respectively.

Q: What do you expect the relationship between age and pay to be? Why?

We now run the following regression:

```
pay.lm<-lm(hourpay~age*genderCat,data=Hourly_Pay)
display(pay.lm)
```

```
## lm(formula = hourpay ~ age * genderCat, data = Hourly_Pay)
##                coef.est coef.se
## (Intercept)     8.41     0.64
## age             0.08     0.01
## genderCatM     -1.56     0.93
## age:genderCatM  0.12     0.02
## ---
```

```
## n = 10458, k = 4
## residual sd = 13.33, R-Squared = 0.04
```

There are a number of problems with the interpretation of the intercept and indeed the coefficients of the predictors:

- The intercept says that a male of age 0 earns 8.41 an hour
- The coefficient of genderCatM (i.e. men vs women) is -1.56 . Because of the interaction interpretation of this coefficient is problematic. We might be tempted to interpret it as meaning that the pay of men is on average £-1.56 pounds lower an hour than that of women. This is incorrect though because of the interaction.
- It actually means that *amongst those of age 0* the pay of men is £-1.56 pounds an hour lower than that of women. That is even more meaningless.

We can make the output more interpretable by first of all *centering* some of the predictors (typically the continuous ones) around their means.

```
cent.age<-with(Hourly_Pay,age-mean(age))
#cent.gender<-with(Hourly_Pay,gender-mean(gender))
pay.lm.cent<-lm(hourpay~cent.age*genderCat,data=Hourly_Pay)
display(pay.lm.cent)
```

```
## lm(formula = hourpay ~ cent.age * genderCat, data = Hourly_Pay)
##                     coef.est coef.se
## (Intercept)         11.88     0.18
## cent.age             0.08     0.01
## genderCatM           3.46     0.26
## cent.age:genderCatM  0.12     0.02
## ---
## n = 10458, k = 4
## residual sd = 13.33, R-Squared = 0.04
```

Note that the residual sd and the R-squared stay the same. That is because centering is a *linear* transformation. The coefficient associated with age does not change, however we can now see that for a person who is 41.7 years old (mean of age) men earn on average 3.46 pounds more than women. This is much more meaningful.

Q: How would you interpret the intercept?

We could also subtract the mean from a binary variable as this would reflect the proportion of e.g. men vs women in the population but in this case we prefer to highlight the difference. We can also on occasion centre the outcome but this is not very common and there needs to be good reason.

Standardising the predictors

If we look at the coefficients, we may be inclined to think that age is a less important predictor than gender because it is smaller in terms of absolute value. It is hard to say for sure though because the *scale* (range) of age is from 16-65 while that of gender is 0-1. A way of making the coefficients more comparable is by *standardising* the predictors i.e. subtracting the mean and dividing by the standard deviation. We'll consider the regression without an interaction for simplicity.

```r
#we use gender here as it is 0,1 and can be manipulated as a number
std.age<-with(Hourly_Pay, (age-mean(age))/sd(age))
std.gender<-with(Hourly_Pay,(gender-mean(gender)/sd(gender)))

pay.lm.std<-lm(hourpay~std.age+std.gender,data=Hourly_Pay)
display(pay.lm.std)
```

```
## lm(formula = hourpay ~ std.age + std.gender, data = Hourly_Pay)
##             coef.est coef.se
## (Intercept) 11.69     0.19
## std.age      1.73     0.13
## std.gender  -3.46     0.26
## ---
## n = 10458, k = 3
## residual sd = 13.35, R-Squared = 0.03
```

Once standardised we see that the impact of gender on hourly pay is larger than that of age. A change in 2 years of (2×1.73=3.46) has approximately the same effect on hourly pay as being a man vs being a woman.

Logarithmic transforms

Logarithmic or log transforms are amongstthe most commonly used transforms. They are called upon when linearity, additivity or constant variance cannot be assumed to hold. They have a useful property which is that they preserve the order of the transformed data and "even" out the data, so that large differences e.g between large and small values in the data become smaller. Common cases where a logarithmic transform should be considered are:

- When the outcome cannot be negative
- When there is a theoretical reason (e.g. economic/scientific theory)
- When the residual plots exhibit a funnel shape

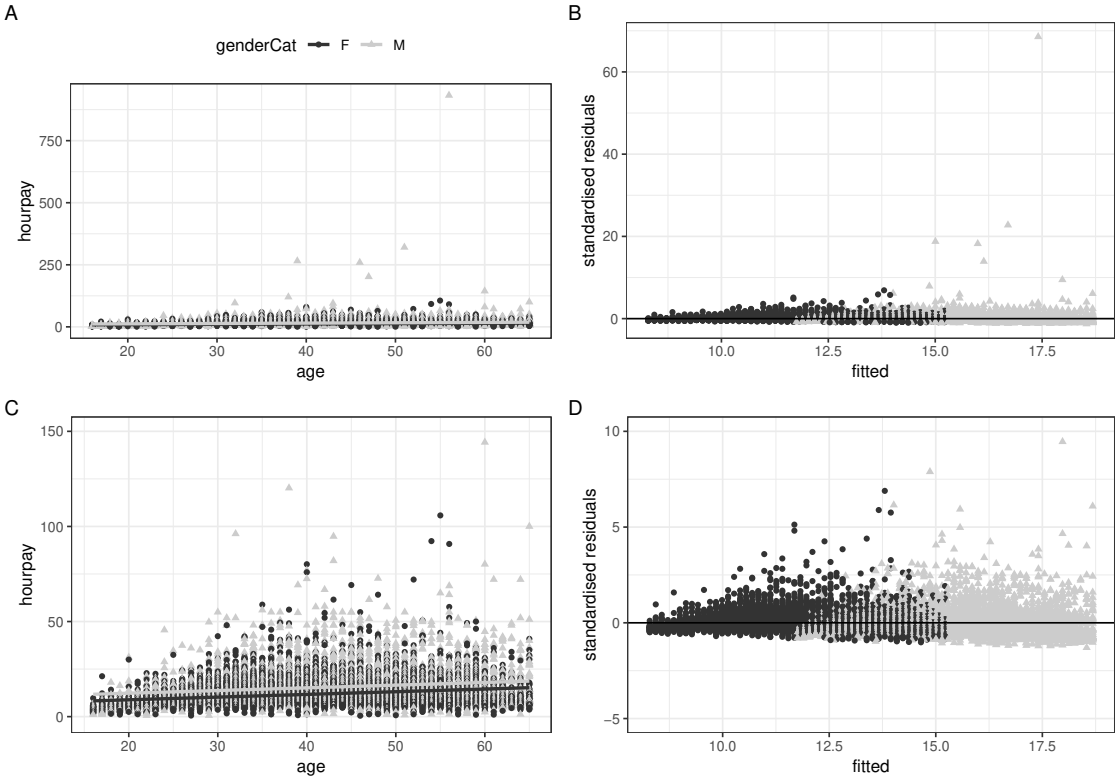
We'll ignore the interaction for now and return to it in the workshop. We now fit the following regression:

$E(\texttt{hourpay}) = \beta_0 + \beta_1 \texttt{age} + \beta_2 \texttt{genderCat}$

```r
pay.lm.0<-lm(hourpay~age+genderCat,data=Hourly_Pay)
display(pay.lm.0)
```

```
## lm(formula = hourpay ~ age + genderCat, data = Hourly_Pay)
##             coef.est coef.se
## (Intercept) 6.03     0.48
## age         0.14     0.01
## genderCatM  3.46     0.26
## ---
## n = 10458, k = 3
## residual sd = 13.35, R-Squared = 0.03
```

Let's look at the scatter and residual plots:



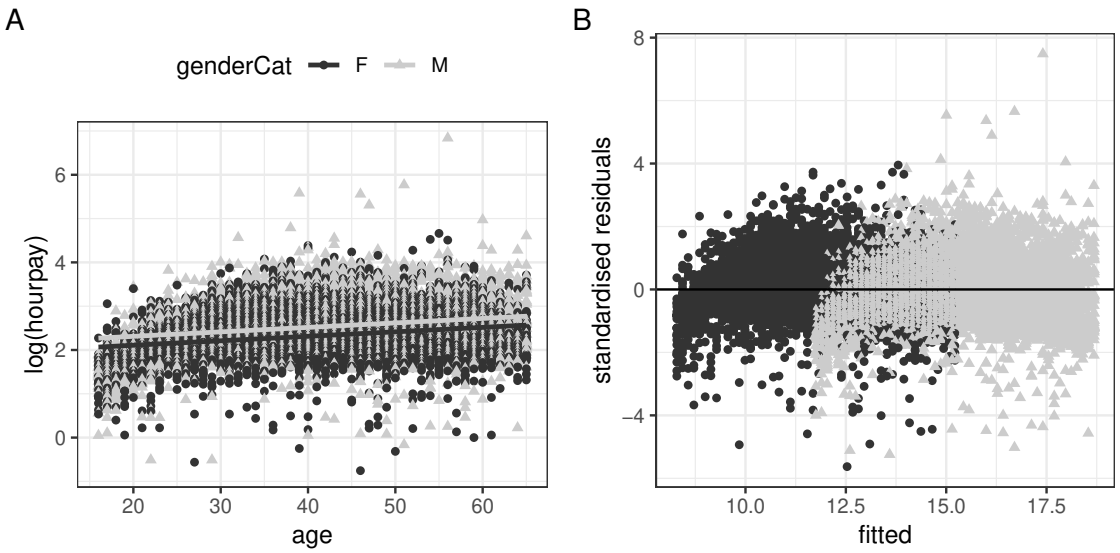
There are quite a few problems highlighted by these plots. First of all in plot A nothing can really be seen, this is also true for the corresponding fitted vs residual plot B. If we remove the most extreme 5 values and re-run the plots we can see that the relationship between `age` and `hourpay` does not look linear in plot C. Furthermore, there is a clear funnel shape in the fitted vs residual plot in D. Note also that the fitted vs residual shows a clear distinction in the predictions between men (in grey) and women (in black).

Q: What do the funnel shape of the residual vs fitted plot means in this data set. Why is it a potential problem?

Given that we have a *funnel shaped plot* in the residual vs fitted plot and the value of hourly pay *can only be positive*, let's try a logarithmic transform on the outcome hourly pay. That means we fit the following regression:

$E(\log(\texttt{hourpay})) = \gamma_0 + \gamma_1 \texttt{age} + \beta_2 \texttt{genderCat}$

The plots look like this:



Not all the problems have gone away. There is still non-linearity, however this is true only for ages below about 30. It is approximately linear for ages over 30 (why could this be?). The fitted versus residual plot has also improved although it is still exhibiting some non-linearity and funnelling.

Q: Why do you think there is a curve shape to the scatterplot? What functional form could be used to address the lack of linearity?

Interpreting the output when using a log transform on the outcome

Applying a log transform on the outcome *changes* the underlying regression assumptions. Without the transform we are saying that there is an *additive* relationship between the outcome and the predictors:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \epsilon_i$$

With a log transform we say it is *additive on the log scale*:

$$log(y_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \epsilon_i$$

Which means *multiplicative* on the original scale:

$$\begin{aligned} y_i &= e^{\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \epsilon_i} \\ &= B_0 \cdot B_1^{x_i 1} \cdot B_2^{x_i 2} \cdot \ldots \cdot B_p^{x_i p} \cdot E \end{aligned}$$

In our case we have

```
pay.lm.1<-lm(log(hourpay)~age+genderCat,data=Hourly_Pay)
display(pay.lm.1)
```

```
## lm(formula = log(hourpay) ~ age + genderCat, data = Hourly_Pay)
##             coef.est coef.se
## (Intercept) 1.90     0.02
```

```
## age         0.01     0.00
## genderCatM  0.20     0.01
## ---
## n = 10458, k = 3
## residual sd = 0.56, R-Squared = 0.08
```

Write down the regression:

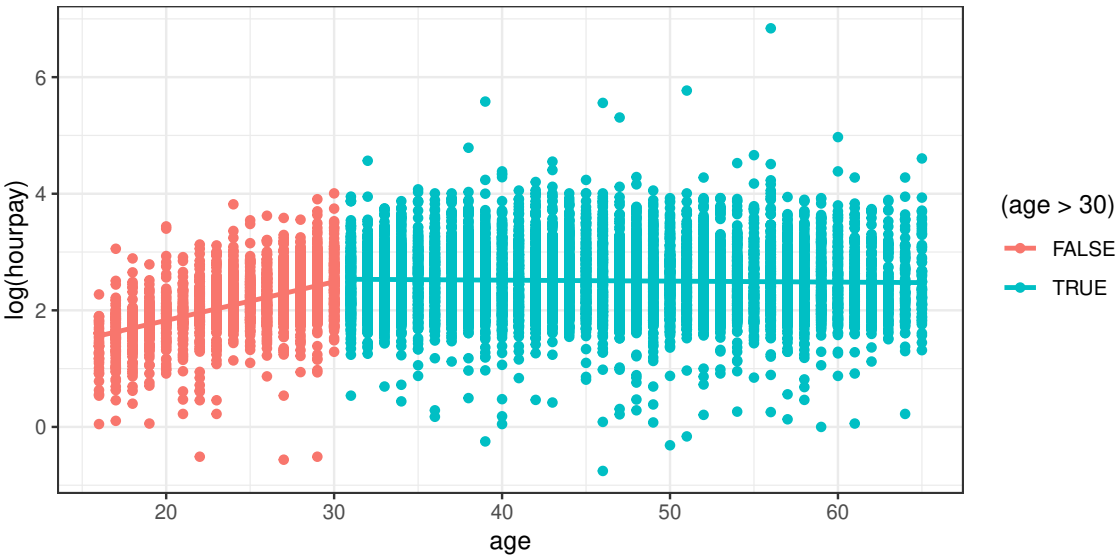
This means that an increase in 1 year of age results in an average increase in 0.01 in *log* hourly pay. This results in an increase by *a factor of* exp(0.01)=1.01 in hourly pay. This is more easily understood as an increase in 1% in hourly play because *multiplying* by 1.01 is equivalent to increasing by 1%.

Q: How do you interpret the coefficient of gender?
On the log scale: A decrease in expected 0.20 in log hourly pay if you are a woman compared to a man
Note that exp(-0.20)=0.82

The residual sd and the R-squared haven't improved very much but that is to be expected as the plots indicate that the fit still isn't very good. We really need to include more predictors and use more complex models to fully analyse these data.

Aside

One way to tackle the non-linearity might be to conduct separate subset analyses, one for people aged below 30 and another for people aged above 30. This could be coded in the regression.

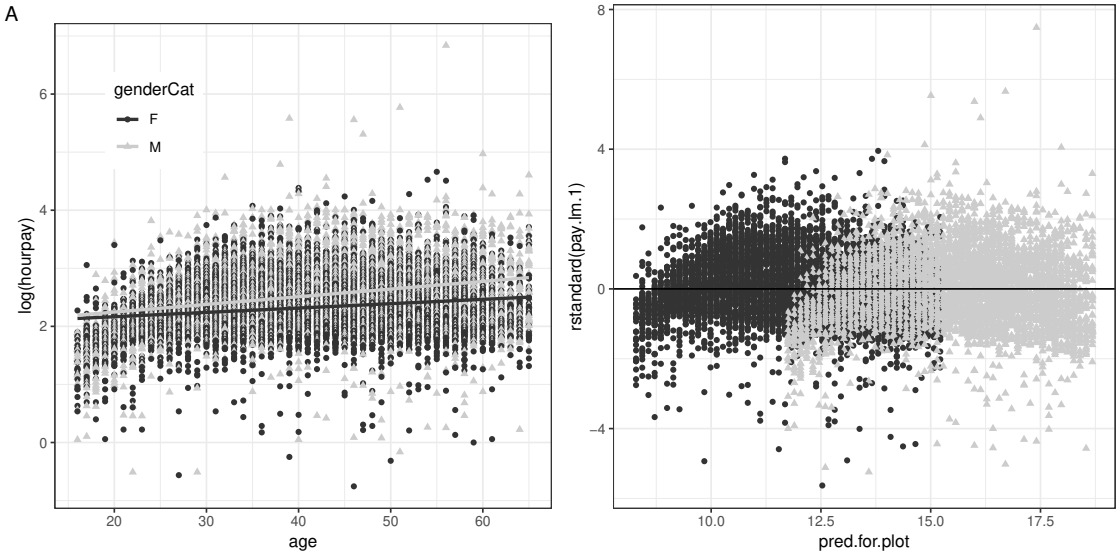


Adding an interaction

What happens when we add the interaction back in?

```
pay.lm.log<-lm(log(hourpay)~age*genderCat,data=Hourly_Pay)
display(pay.lm.log)
```

```
## lm(formula = log(hourpay) ~ age * genderCat, data = Hourly_Pay)
##                coef.est coef.se
## (Intercept)     2.02     0.03
## age             0.01     0.00
## genderCatM     -0.03     0.04
## age:genderCatM  0.01     0.00
## ---
## n = 10458, k = 4
## residual sd = 0.55, R-Squared = 0.08
```



The best way to interpret all the coefficients in the light of the interaction is to write down two separate equations and apply what we've learnt so far.

Write down the two equations corresponding to the two values of gender

Using the two regressions above:

Q: For each coefficient except the intercept write down the:
1. coefficient
2. what % change in the hourly pay does it imply?

0 values and log transforms
Log transforms cannot be applied to variables with 0 or negative values. Think carefully whether you should change these data (e.g. by adding a fixed amount) in order to use this transform. One common solution is to add 0.5 to 0 values. Doing this needs to be justified.

Other possible transforms on the outcome:

Square root

This shrinks high values but less than the log transform. However it is very hard to interpret coefficients.

Box Cox

It often helps with the diagnostic statistics but is complex and again leads to very hard to interpret coefficients.

Workshop 6: Transforms and non-linear relationships

Learning outcomes

How to apply transforms in R

Assessing the effect of transforms on residual plots/model

Understanding how transforms affect interpretability of regression results

Packages

```
library(arm)
library(ggplot2)
library(gridExtra)
library(tidyr)
```

Transforms in R

In this workshop we will try a number of different transforms. Not all the transforms are well motivated in the examples below but I apply them anyway for teaching purposes. If you choose to use transforms it is important that you justify carefully why you use them. Valid reasons include:

1. To improve interpretation
2. To improve prediction/model fit
3. Because it makes sense from a theoretical point of view

For point two you need to carefully consider whether the transforms improve model fit enough to compensate for the loss of interpretability, although this is less of a concern if the main aim of the analysis is prediction.

We'll use the Hourly Pay data again but with more predictors. We will also reduce the data-set to avoid values were `hourpay` are above £150 per hour.

```
Hourly_Pay<-read.csv("Large_Hourly_Pay.csv",header = TRUE)
Hourly_Pay_NoRich<-subset(Hourly_Pay,hourpay<=150)
summary(Hourly_Pay_NoRich)
```

Some data re-arranging

If nothing is changed R chooses the baseline category for categorical variables as the first in alphabetical order. This is often not the best choice. Better choices are

a) the most common,
b) the one of most interest for comparison.

For our data we need to choose baseline levels for the categorical values and make sure R knows about it. We pick the level with the most values. We find out which these are by looking at `summary()`. We'll also rename the gender variable so that instead of 0,1 it is to "M" and "F" respectively.

```
Hourly_Pay_NoRich$OwnHouse<-relevel(Hourly_Pay_NoRich$OwnHouse,ref="Own")
Hourly_Pay_NoRich$Ethnicity<-relevel(Hourly_Pay_NoRich$Ethnicity,ref="White")
Hourly_Pay_NoRich$HighestQual<-relevel(Hourly_Pay_NoRich$HighestQual,ref="DegreeHE")
Hourly_Pay_NoRich$gender<-ifelse(Hourly_Pay_NoRich$gender==0,"M","F")
```

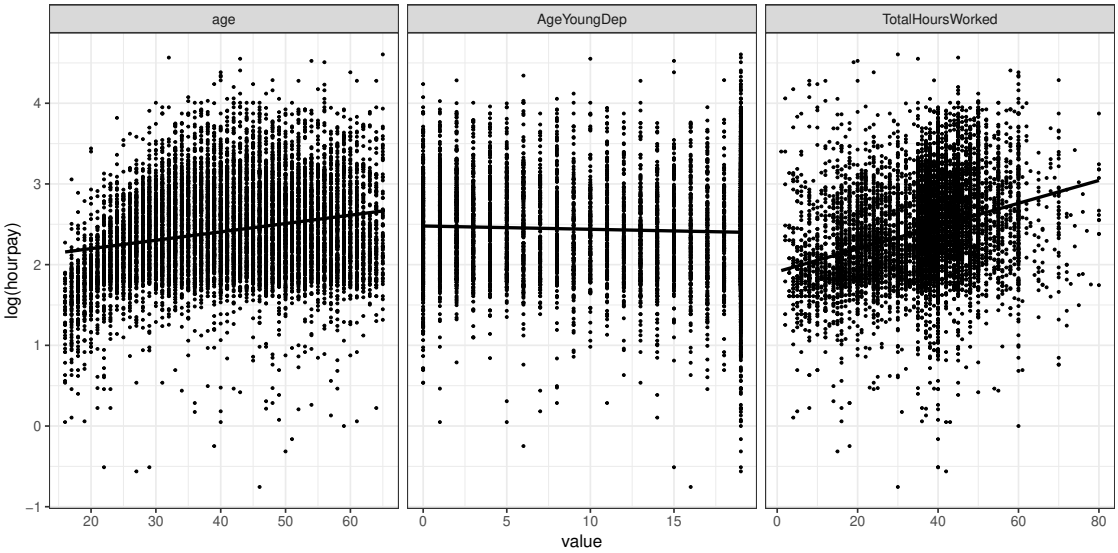
The `relevel()` function takes the categorical predictor as the first argument, `ref=` assigns the baseline level.

Before going any further let's think about what you expect the signs associated with the coefficients for the various predictors:

Name	sign	Name	sign	Name	sign
Rent vs Own		Chinese vs White		Alevel vs DegreeHE	
Other vs Own		Afric/Carib vs White		GCSE vs DegreeHE	
Woman vs Man		ISC vs White		None vs ALevGCSE	
age		TotalHoursWorked		AgeYoungDep	

Plot the continuous predictors against the outcome.

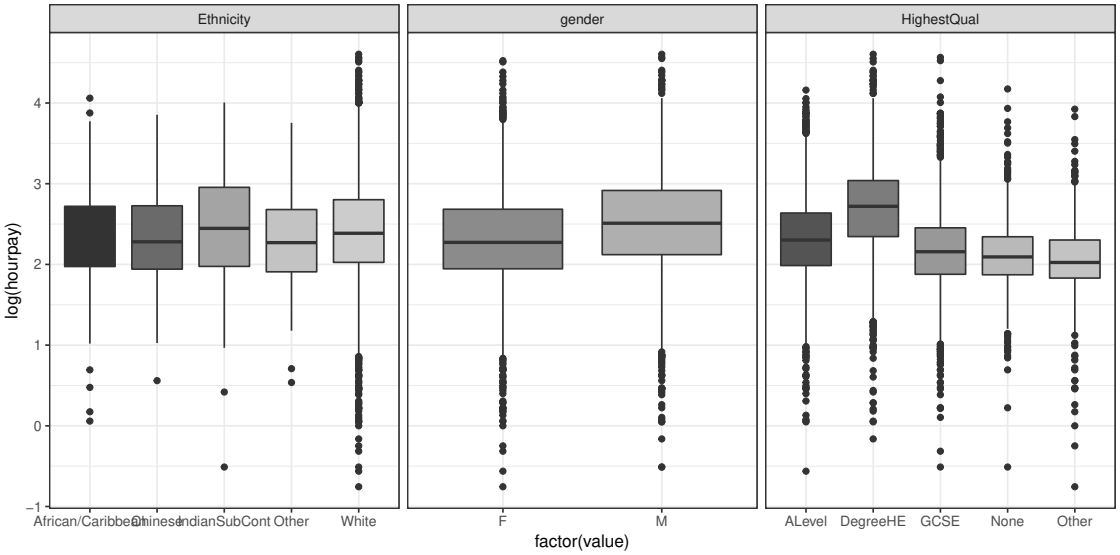
```
# Using facet_wrap to display the scatterplots line.1
special.dat <- gather(Hourly_Pay_NoRich[, c(3, 4, 6, 7), ], -hourpay, key = "var",
    value = "value")
# line.2
p1 <- ggplot(special.dat, aes(x = value, y = log(hourpay))) + geom_point(size = 0.5) +
    geom_smooth(method = "lm", se = FALSE, color = "black") + facet_wrap(~var,
    scales = "free_x")
# for book
p1 <- p1 + theme_bw() + scale_color_grey() + theme(legend.position = "none")
# line.3
p1
```



line.1: Defines the data set to pass to the plot. If you type `colnames(Hourly_Pay_NoRich)` you'll see that columns 3,4,6 and 7 correspond to the numeric (and possibly continuous) variables in the dataset. `gather()` is a clever function from the "tidyr" package that stacks the predictors on top of each other in one variable, called `value`. Then `R` treats the different "levels" of value, called `variable` as it would e.g. levels in a categorical variable. Have a look at `special.dat` to see exactly what this means. line.2: Defines the x and y's in the `ggplot()` line.3: Plots by setting x=value as defined above, `facet_wrap()` make the three plots of log(`hourpay`) against the variables in the data frame that aren't y or the shape/colour variables by using `free_x`.

Also the boxplots for the categorical predictors.

```
# Define the categorical variables as well as hourpay
special.dat <- gather(Hourly_Pay_NoRich[, c(2, 5, 6, 8)], key = "variable",
    value = "value", -hourpay)
# plots the boxplots using 'free_x'
p1 <- ggplot(special.dat, aes(x = factor(value), y = log(hourpay), fill = factor(value))) +
    geom_boxplot() + facet_wrap(~variable, scales = "free_x")
# for book
p1 <- p1 + theme_bw() + theme(legend.position = "none") + scale_fill_grey()
p1
```



We get a warning here, ignore it. It has to do with the fact that the x's are of different types.

Q: Do you notice any patterns in the plots? Discuss.

Based on the plots, which predictors do you think have the strongest marginal association with log(hourpay)?

The regression with all the predictors is:

```
all.lm<-lm(hourpay~.,data=Hourly_Pay)
#shortcut if you want to use all the predictors in the dataset
all.lm<-lm(hourpay~age+TotalHoursWorked+AgeYoungDep+gender
           +OwnHouse+HighestQual+Ethnicity,data=Hourly_Pay_NoRich)
display(all.lm)
```

```
## lm(formula = hourpay ~ age + TotalHoursWorked + AgeYoungDep +
##     gender + OwnHouse + HighestQual + Ethnicity, data = Hourly_Pay_NoRich)
```

```
##                              coef.est coef.se
## (Intercept)                   8.68     0.37
## age                           0.14     0.01
## TotalHoursWorked              0.10     0.01
## AgeYoungDep                  -0.13     0.01
## genderM                       2.37     0.16
## OwnHouseOther                -4.60     0.89
## OwnHouseRent                 -2.18     0.18
## HighestQualALevel            -5.09     0.19
## HighestQualGCSE              -6.40     0.20
## HighestQualNone              -8.02     0.29
## HighestQualOther             -8.13     0.34
## EthnicityAfrican/Caribbean   -1.50     0.52
## EthnicityChinese             -1.18     0.66
## EthnicityIndianSubCont       -0.20     0.44
## EthnicityOther                0.14     0.55
## ---
## n = 10372, k = 15
## residual sd = 7.44, R-Squared = 0.27
```

Q: In your groups interpret the coefficients. Are there any unexpected values (by which I mean are there any signs that are unexpected or predictors you thought would have significant coefficients that do not?

Let's standardise the continuous predictors to get a better idea of their relative importance. First as we need to perform the same task on 3 predictors let's write a function to do it:

```
standardise.fn<-function(v){(v-mean(v))/(sd(v))}
```

Now let's apply our function.

```
std.age<-standardise.fn(Hourly_Pay_NoRich$age)
std.THW<-standardise.fn(Hourly_Pay_NoRich$TotalHoursWorked)
std.AYD<-standardise.fn(Hourly_Pay_NoRich$AgeYoungDep)
```

And run the regression

```
std.all.lm<-lm(hourpay~std.age+std.THW+std.AYD+gender
               +OwnHouse+HighestQual+Ethnicity,data=Hourly_Pay_NoRich)
display(std.all.lm)
```

```
## lm(formula = hourpay ~ std.age + std.THW + std.AYD + gender +
##     OwnHouse + HighestQual + Ethnicity, data = Hourly_Pay_NoRich)
##                             coef.est coef.se
## (Intercept)                 16.47     0.14
## std.age                      1.73     0.08
## std.THW                      1.28     0.08
## std.AYD                     -0.85     0.08
## genderM                      2.37     0.16
## OwnHouseOther               -4.60     0.89
## OwnHouseRent                -2.18     0.18
## HighestQualALevel           -5.09     0.19
## HighestQualGCSE             -6.40     0.20
## HighestQualNone             -8.02     0.29
## HighestQualOther            -8.13     0.34
## EthnicityAfrican/Caribbean -1.50     0.52
## EthnicityChinese            -1.18     0.66
## EthnicityIndianSubCont      -0.20     0.44
## EthnicityOther               0.14     0.55
## ---
## n = 10372, k = 15
## residual sd = 7.44, R-Squared = 0.27
```

Q: Which of the continuous predictors appears to be most important and why?

Would you remove any predictors? How can you find out if any of the categorical predictors should be removed?

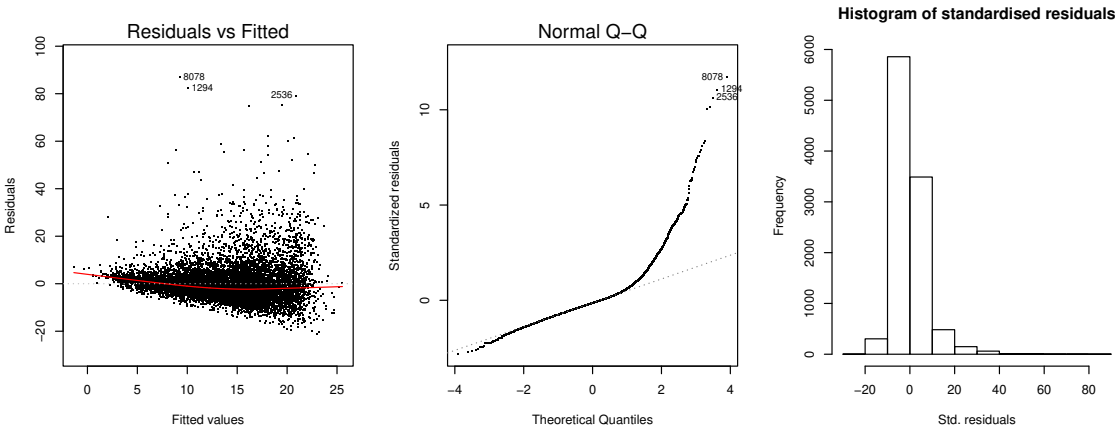
```
Anova(all.lm)
```

```
## Anova Table (Type II tests)
##
## Response: hourpay
##                   Sum Sq    Df  F value  Pr(>F)
## age                26422     1 477.7901 < 2e-16 ***
## TotalHoursWorked   14087     1 254.7337 < 2e-16 ***
## AgeYoungDep         6875     1 124.3175 < 2e-16 ***
## gender             12275     1 221.9766 < 2e-16 ***
## OwnHouse            9635     2  87.1162 < 2e-16 ***
## HighestQual       100069     4 452.3911 < 2e-16 ***
## Ethnicity            640     4   2.8948 0.02082 *
## Residuals         572742 10357
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual plots

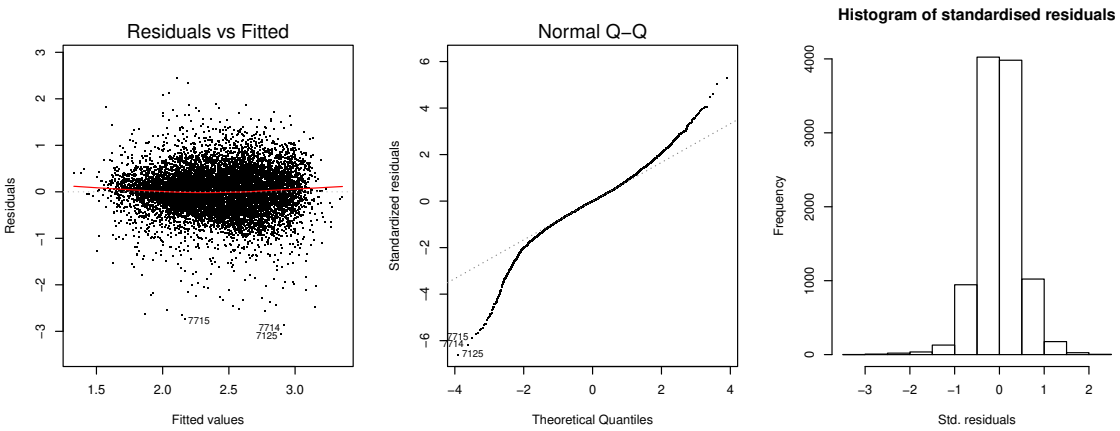
Let's look at the residual plots:

```
par(mfrow=c(1,3))
plot(all.lm,which=c(1,2),cex=0.7,pch=".")
hist(all.lm$residuals,main="Histogram of standardised residuals", xlab="Std. residuals")
```



There is definitely a "funnel" shape in the residual vs fitted plot as well as evidence of skewed residuals. Let's try a log transform on the outcome hourpay:

```
log.lm<-lm(log(hourpay)~.,data=Hourly_Pay_NoRich)
par(mfrow=c(1,3))
plot(log.lm,which=c(1,2),cex=0.7,pch=".")
hist(log.lm$residuals,main="Histogram of standardised residuals", xlab="Std. residuals")
```



Let's look at `display()`:

```
display(log.lm)
```

```
## lm(formula = log(hourpay) ~ ., data = Hourly_Pay_NoRich)
##                          coef.est coef.se
## (Intercept)               2.01     0.02
## OwnHouseOther            -0.30     0.06
## OwnHouseRent             -0.15     0.01
## genderM                   0.14     0.01
## age                       0.01     0.00
## AgeYoungDep              -0.01     0.00
```

```
## EthnicityAfrican/Caribbean -0.11     0.03
## EthnicityChinese           -0.09     0.04
## EthnicityIndianSubCont     -0.04     0.03
## EthnicityOther              0.00     0.03
## TotalHoursWorked            0.01     0.00
## HighestQualALevel          -0.35     0.01
## HighestQualGCSE            -0.47     0.01
## HighestQualNone            -0.58     0.02
## HighestQualOther           -0.62     0.02
## ---
## n = 10372, k = 15
## residual sd = 0.46, R-Squared = 0.34
```

Based on the output of `display()` and the residual plots, let's compare the two models.

Diagnostics

- $\mathbf{R}^2$ is 0.34 in log model vs 0.27 in the raw `hourpay` model.
- **Residual sd**: both are small relative to the size of the range.
  - for the log model: 0.46. This needs to be compared with the range of log(`hourpay`) which is -0.76 to 4.61 with a width of 5.37
  - for the raw model: 7.44. This needs to be compared range of `hourpay` is 0.47 to 100.00
- **Significance of coefficients**: In the log model, ethnicity is more significant, with all the coefficients associated with it being significant or borderline non-significant at the 5% level.
- **The Adjusted R-squared** (from `summary()`) shows that the log model is a tiny bit better
  - for the log model: 0.34 which is almost identical to the R^2 (so no hint of multicollinearity)
  - for the original model: 0.27 which is almost identical to the R^2
- **F-statistic p-value** (from `summary()`) is very low for both models.
  - for the log model: p-value: $<$ 2.2e-16
  - for the original model: p-value: $<$ 2.2e-16

We conclude that the fit of the log transformed model is *slightly* better than that of the original model. We would prefer the log transformed model, however we *need to acknowledge* that it isn't a very good model (low R^2).

Residual Plots

- The fitted vs residual plot: The plot in the original model exhibits a clear funnel shape. The fitted vs residual plot in the logged model looks far more like random scatter. Independence of the error terms holds
- The Q-Q plot in the original model exhibits strong skew (it is bow shaped) with a lot of deviation, however, the Q-Q plot in the logged model exhibits *kurtosis* (it is s-shaped). However this is less extreme than the skewness of the original model. Normality holds approximately
- The histograms tell similar stories.

Conclusions:

Without using more complex models we cannot do better (and even with more complex models we may fail to do better). This means that we can interpret the outcomes and draw conclusions about the relationships between pay and the predictors using the model, but that we *need to say that they are based on a model that is not entirely satisfactory.*

Q: Interpret the coefficients of "gender", "TotalHoursWorked" and "EthnicityChinese" in the log model.

Quadratic transform

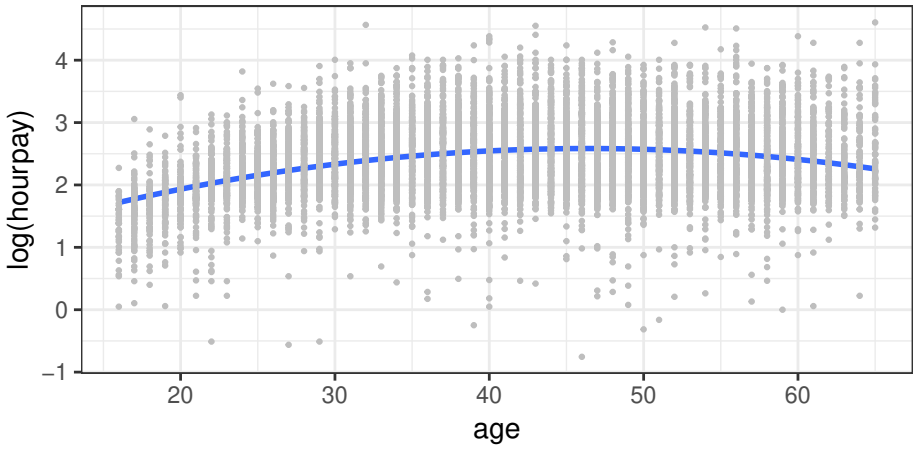
Let's try a quadratic transform of *age* to account for the curvature.

```
age2<-Hourly_Pay_NoRich$age*Hourly_Pay_NoRich$age
quad.age.lm<-lm(log(hourpay)~age+age2+TotalHoursWorked+AgeYoungDep+gender
           +OwnHouse+HighestQual+Ethnicity,data=Hourly_Pay_NoRich)
display(quad.age.lm)
```

```
## lm(formula = log(hourpay) ~ age + age2 + TotalHoursWorked + AgeYoungDep +
##     gender + OwnHouse + HighestQual + Ethnicity, data = Hourly_Pay_NoRich)
##                              coef.est coef.se
## (Intercept)                   1.17     0.05
## age                           0.06     0.00
## age2                          0.00     0.00
## TotalHoursWorked              0.01     0.00
## AgeYoungDep                   0.00     0.00
## genderM                       0.15     0.01
## OwnHouseOther                -0.29     0.05
## OwnHouseRent                 -0.16     0.01
## HighestQualALevel            -0.32     0.01
## HighestQualGCSE              -0.45     0.01
## HighestQualNone              -0.56     0.02
## HighestQualOther             -0.60     0.02
## EthnicityAfrican/Caribbean   -0.12     0.03
## EthnicityChinese             -0.09     0.04
## EthnicityIndianSubCont       -0.05     0.03
## EthnicityOther                0.01     0.03
## ---
## n = 10372, k = 16
## residual sd = 0.46, R-Squared = 0.36
```

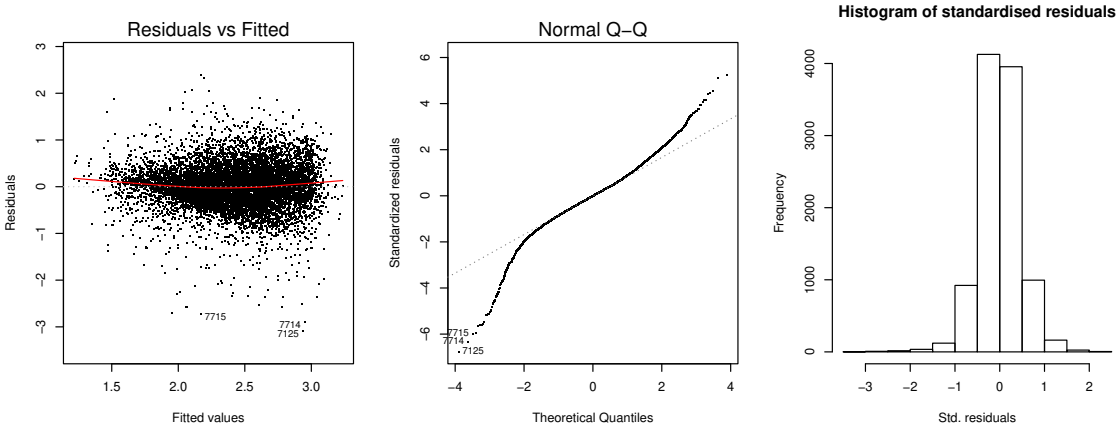
What does it look like?

```
#for geom_line to put the quadratic line
pred.for.plot<-fitted(quad.age.lm)
#plot of raw data
p1<-ggplot(data=Hourly_Pay_NoRich, aes(x=age, y=log(hourpay)))
#using geom_line to fit a line through the fitted points (i.e. the expected line)
p1<-p1+geom_smooth(method="lm", formula=y~poly(x,2), size = 1)
#not essential, for the book
p1<-p1+theme_bw()+ scale_colour_grey()+geom_point(color="grey", size=0.5)
p1
```



What about the corresponding residual plots?

```
#Residual plots easier with plot()
par(mfrow=c(1,3))
plot(quad.age.lm,which=c(1,2),cex=0.7,pch=".")
hist(quad.age.lm$residuals,main="Histogram of standardised residuals", xlab="Std. residuals")
```



Q: Compare the logged model with the logged model and quadratic age. Which do you prefer? Bear in mind that adding a quadratic term can considerably complicate the interpretation.

Log transform on the predictor

Let's try to log transform age. This might deal with the curvature of the plot.

```
log.age.lm<-lm(log(hourpay)~log(age)+TotalHoursWorked+AgeYoungDep+gender
           +OwnHouse+HighestQual+Ethnicity,data=Hourly_Pay_NoRich)
```

```
display(log.age.lm)
```

Produce residual plots and a scatterplot for this model.

Recoding categorical variables

Sometimes we want to recode categorical variables so they have fewer levels. This is especially useful if you have a lot of categorical variables with many levels. This makes interpretation easier and often solves the problem of one significant level and many non-significant levels.

Consider HighestQual:

```
with(Hourly_Pay_NoRich, levels(HighestQual))
```

```
## [1] "DegreeHE" "ALevel"   "GCSE"     "None"     "Other"
```

We may be interested in distinguishing between those with a Degree, and A-level and everyone else. We use the `Recode` function from the `car` library

```
new.HQ<-with(Hourly_Pay_NoRich,Recode(HighestQual,"c('GCSE','None','Other')='Other'"))
head(Hourly_Pay_NoRich$HighestQual)
head(new.HQ)
new.HQ<-relevel(new.HQ,ref="DegreeHE")
```

Now what happens when we re-run the regression?

```
recode.lm<-lm(log(hourpay)~age+TotalHoursWorked+AgeYoungDep+gender
           +OwnHouse+new.HQ+Ethnicity,data=Hourly_Pay_NoRich)
display(recode.lm)
```

```
## lm(formula = log(hourpay) ~ age + TotalHoursWorked + AgeYoungDep +
##     gender + OwnHouse + new.HQ + Ethnicity, data = Hourly_Pay_NoRich)
##                             coef.est coef.se
## (Intercept)                  2.03     0.02
## age                          0.01     0.00
## TotalHoursWorked             0.01     0.00
## AgeYoungDep                 -0.01     0.00
## genderM                      0.13     0.01
## OwnHouseOther               -0.30     0.06
## OwnHouseRent                -0.16     0.01
## new.HQALevel                -0.35     0.01
## new.HQOther                 -0.51     0.01
## EthnicityAfrican/Caribbean -0.11     0.03
## EthnicityChinese            -0.10     0.04
## EthnicityIndianSubCont      -0.05     0.03
## EthnicityOther              -0.01     0.03
## ---
## n = 10372, k = 13
## residual sd = 0.47, R-Squared = 0.34
```

R tips: shortcuts and changing the dataframe

If you have a lot of variables and want to drop one you can use the following:

```
display(lm(log(hourpay)~.,data=Hourly_Pay_NoRich))
display(lm(log(hourpay)~.-Ethnicity,data=Hourly_Pay_NoRich))
```

If you want to be able to use this with transformed (e.g. standardised variables) then

```
new.HPW<-Hourly_Pay_NoRich
new.HPW$age<-std.age
new.HPW$TotalHoursWorked<-std.THW
new.HPW$AgeYoungDep<-std.AYD
new.HPW$HighestQual<-new.HQ
```

We can then re-run the regression:

```
new.lm<-lm(log(hourpay)~.-Ethnicity,data=new.HPW)
display(new.lm)
```

```
## lm(formula = log(hourpay) ~ . - Ethnicity, data = new.HPW)
##                    coef.est coef.se
## (Intercept)         2.66     0.01
## OwnHouseOther      -0.30     0.06
## OwnHouseRent       -0.17     0.01
## genderM             0.13     0.01
## age                 0.13     0.00
## AgeYoungDep        -0.06     0.00
## TotalHoursWorked    0.11     0.01
## HighestQualALevel -0.34     0.01
## HighestQualOther  -0.51     0.01
## ---
## n = 10372, k = 9
## residual sd = 0.47, R-Squared = 0.34
```

Other transforms

Other transforms are the inverse, square root or the Box-Cox transform. They all make interpretation much harder but should be considered in situations where prediction is more important and small changes to prediction can result in large gains.

You could try the Omitted variable exercise at the back of the book now

Lecture 7: Prediction and Validation

Learning outcomes

Understand how to predict using regressions

Understanding the principle of cross-validation for model fit and prediction

Model building: at the end of this week you will have all the basic tools to run an analysis using a linear model. Refer to the back of the book for the chapter on Model Building.

Why regression (again)?

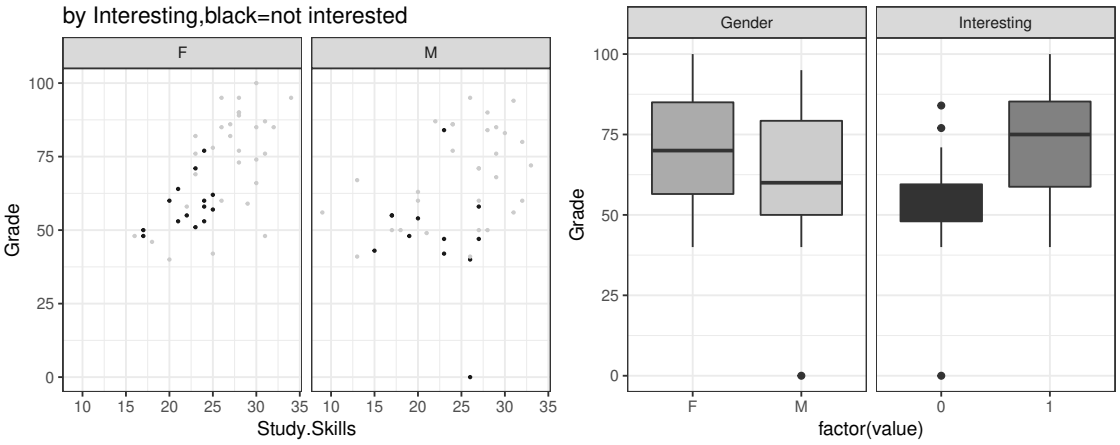
The goals of regression are broadly two (and most analyses have both elements):

- One aim is *to understand the relationships between predictors (explanatory variables) and outcomes*. This is important for decision making: e.g. if we see that women earn less than men for the same types of jobs the maybe it is necessary to make laws to ensure this does not happen (sadly there are laws but it still happens!)
- Another aim is to *predict values for new data*. For example a student might want to predict, given the values of their predictor variables what grade they are expected to get on their stats exam using data from students in previous years. The use of statistical models in finance and climate have almost exclusively the goal of prediction of e.g. future prices and volatility, the weather in the next week. In these cases we are interested in *out of sample* prediction.

Even when prediction is not the principal aim of a regression, it can be useful to use it as a tool to *validate* the model. I.e. if a model is good at predicting then it is telling us something true about the underlying relationships between the variables. In these cases we often use a tool called *cross-validation*. Simply put, cross-validation divides the data into two parts, and uses one – termed the training set – to fit a model and then predict the other part – termed the test set. How well the model predicts can be seen as an indication of how good the model is.

Prediction

Let's look at the Study Habits data again:



Lets look at the model with Gender, Interesting and Study.Skills

```
grade.lm.3<-lm(Grade~Gender+Interesting+Study.Skills, data=Study_habits)
display(grade.lm.3)
```

```
## lm(formula = Grade ~ Gender + Interesting + Study.Skills, data = Study_habits)
##              coef.est coef.se
## (Intercept)  30.06     8.51
## GenderM      -5.70     3.28
## Interesting  13.79     3.77
## Study.Skills  1.18     0.35
## ---
## n = 86, k = 4
## residual sd = 15.03, R-Squared = 0.34
```

Q: Using the model above, calculate by hand, what the expected `Grade` is for a woman student who in interested in statistics and has `Study.Skills` of 30.

Q: Calculate the expected `Grade` for a man with `Study.Skills` of 20 who is not interested in statistics.

The values we've considered are *in-sample*. This means that the values for the predictors for both points are within the observed ranges for these predictors. The range of `Study.Skills` is 9-34 and we've observed both men and women, both interested and uninterested. Because the points are in-sample the model is likely do to well (or at least as well as it can as it is not a great model).

If we want to predict for a new student who is female, has study skills 10 and is interested in statistics then we are moving further away from the main bulk of the data. Compare the location along the x-axis of this point in the scatterplot to see this.

We get `R` to do it: Create a data frame with the new values

```
x.in.sample<-data.frame(Gender=c("F","M"),Interesting=c(1,0),Study.Skills=c(30,20))
x.in.sample
```

```
##   Gender Interesting Study.Skills
## 1      F           1           30
## 2      M           0           20
```

```
x.out.of.sample<-data.frame(Gender=c("M"),Interesting=c(0),Study.Skills=c(10))
x.out.of.sample
```

```
##   Gender Interesting Study.Skills
## 1      M           0           10
```

Use the `predict` function.

1. The first argument is the name of the linear model you want to predict from.
2. The second is the data frame of the values of the predictor variables you want to predict for. `interval="prediction"` says to use the predictive interval. You could use `interval="confidence"` but it makes more sense to ask for confidence intervals for parameters.

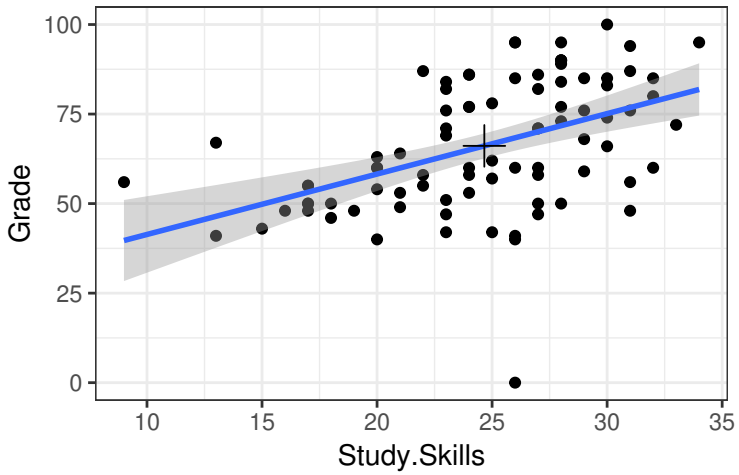
```
predict(grade.lm.3,x.in.sample,interval="predict", level=0.95)
```

```
##        fit      lwr       upr
## 1 79.39353 48.98088 109.80617
## 2 48.05314 17.37614  78.73015
```

```
predict(grade.lm.3,x.out.of.sample,interval="prediction", level=0.95)
```

```
##       fit      lwr      upr
## 1 36.2058 4.575977 67.83563
```

Add the points to the plot below and it becomes clear which points are estimated with less certainty and why.



The bands in grey show the prediction interval. We can see that close to the mean of `Study.Skills`/`Grade` (where the cross is) at 24.7,66.1 this is narrowest.

More Out of sample prediction

The data we have looked at so far is for the 2012 and 2013 cohorts of students. We also have data on the 2015 cohort. Let's see how well the model predicts their grades.

Load the new 2015 cohort data:

```
new.Study_Habits<-read.csv("Stats_study_habits.new.csv",header=T)
```

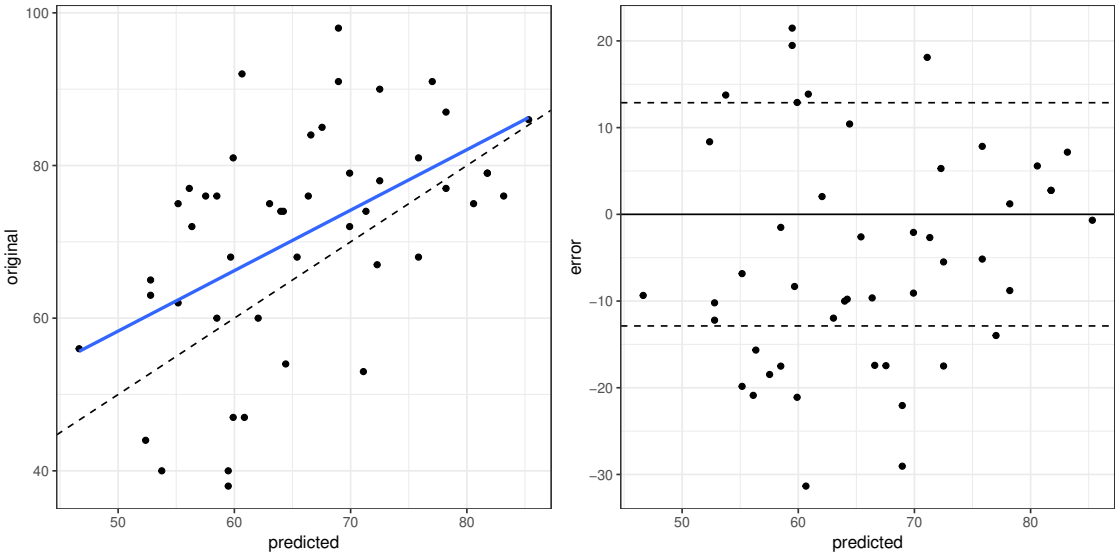
Use it to predict the grades for the 2015 cohort.

```
predict.2015<-predict(grade.lm.3,new.Study_Habits)
```

We can assess how good the predictions in two main ways: One is to look at some plots, the second is to evaluate the mean square predictive error(MSPE). We'll look at the plots only for the study habits data as they are relatively small. When we tackle the hourly pay data next, we'll also look at the MSPE.

To see how good the predictions are visually, we create a new data set with the predictions using the model, the actual data from 2015 and the error which is the difference between the two. These help us to create the following plots:

1. The plot of the predicted vs the original points
2. The plot of the predicted points vs the prediction error associated with these points



1. On the left hand side the solid line is the diagonal and the dotted line is the regression of real vs predicted grade.
2. On the right hand side the dotted lines are ± the standard error of the prediction error.
3. There is quite a bit of variability. Note that the regression itself is not a great fit in the first place (it has an R^2 of 0.36)

Q: Explain why we plot the line with intercept 0 and gradient 1 on the left hand side plot.

Q: Explain how to use the right hand plot.

The predictions are acceptable but not great as can be seen in both of the plots.

Cross-validation

Cross-validation is a very important topic in machine learning. Machine learning is a branch of statistics that focuses mostly on prediction and therefore diagnostic tools that help quantify how well a model is at predicting are essential.

The basic idea is as follows:

1. Take a random % (e.g. 90%) of the data. These are called the *training set*. The remaining % (e.g. 10%) are called the *test set*.
2. Fit a model using the training set data only.
3. Using the model and the values of the predictors for the test set predict the outcome for the test set. These are your predicted outcomes.

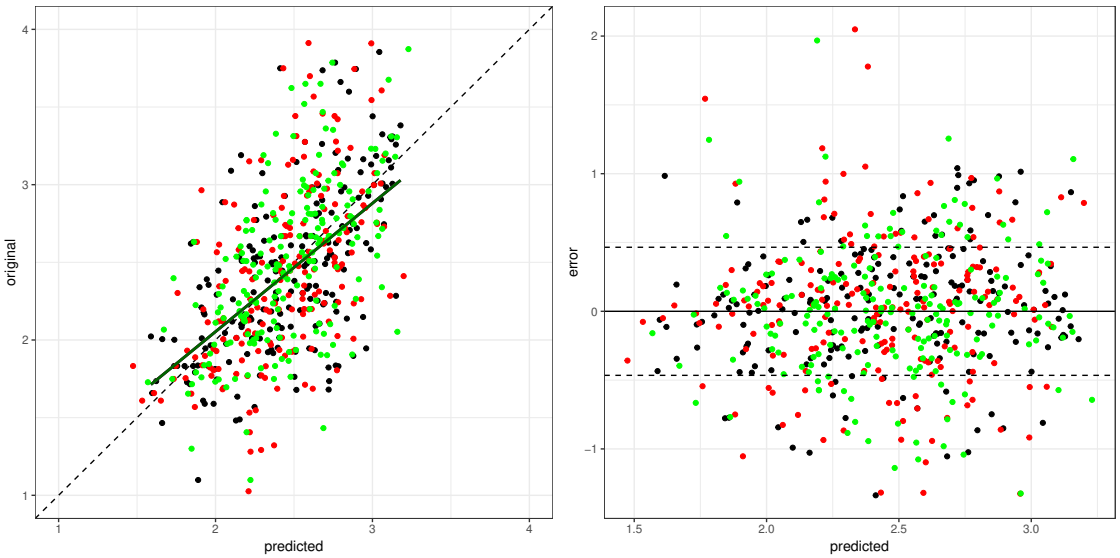
4. Finally, compare the predicted outcomes to the observed outcome in the test set.

You need to do this for more than one 90/10 split because one split could by chance be unrepresentative. You should also try different splits such as a 80/20 or 70/30 split.

The smaller the size of the training set the more variable the results (i.e. the larger the standard errors of the regression coefficients) but the easier it is to see whether the model is doing badly as the model will be predicting a larger outcome sample.

Let's try some cross-validation using the data on hourly pay, however we'll choose a random sample of 2000 points as otherwise it becomes hard to see what is happening in the plots.

We'll create 3 10/90 splits: In order to do cross-validation we need a random sample of 10%: `0.9*nrow(Hourly_pay)` .



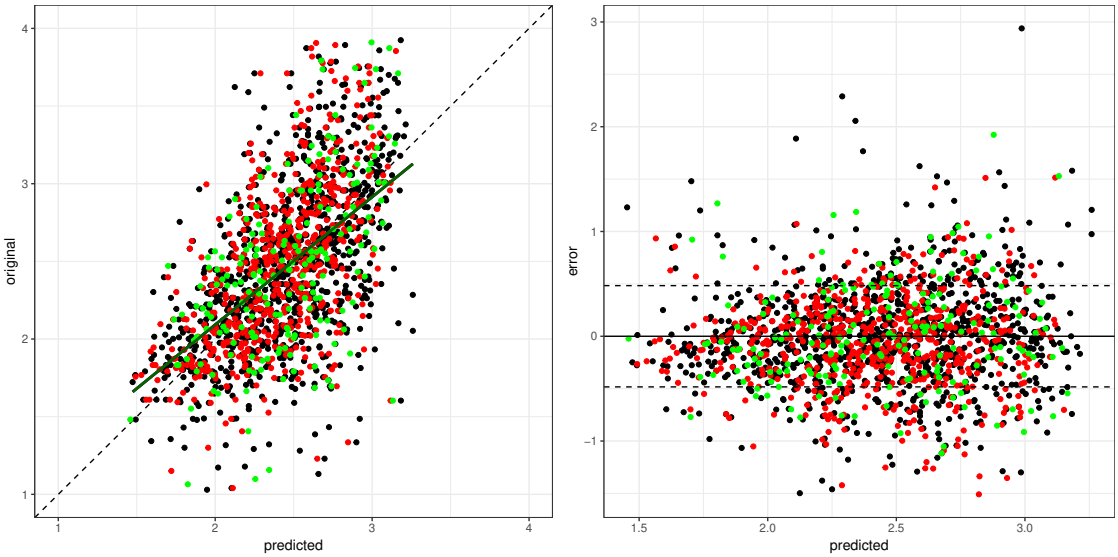
Note I didn't use a black and white colour scale in these plots as it becomes hard to see anything. This is why the plots above aren't very informative. The plots in the slides and in the code are.

We can see that there is a lot of variation in the plots across all three iterations. Specifically the variation is in the y-direction in the predicted vs original plot, indicating that there is more variability in the original `hourpay` than the model is accounting for. This is not great for prediction. On the plus side, the regression of the original `hourpay` on the predicted `hourpay` is very close to the line meaning that there is no non-linear trend that the predictions are not picking up. For the predicted vs error plot, there are a lot of points beyond the one standard deviation lines but the error is not increasing with the size of the predictions.

We can also see that while there is some variability in the predictions for different training/test set splits, they are broadly similar in terms of spread and it is impossible to tell the regression lines in the predicted vs original plots apart.

Using different splits

Let's see what the effect is of having different splits. We'll try 50/50, 70/30 and 90/10 splits. I don't recommend the first one, but it is interesting to see what happens.



We can see that there is a bit less scatter for the green/light (90/10) than the red/middle (70/30) than the black (50/50) points.

Mean square predictive errors

One way of assessing how good a model is, is to compare the mean square of the predictive errors for in and out of sample predictions. The MSPE=$\frac{\sum_1^n e_i^2}{n}$ where e_i is the prediction error for the ith point and n is the size of the training set for in sample prediction and the test set for out of sample predictions. We expect the model to do better in the in sample predictions (i.e. the fitted values) than in the out of sample predictions. When the difference between the two is small, then the model is doing as well in the out-of-sample predictions as it is in the in-sample prediction. Bear in mind that for a poor model both in and out of sample predictions may be poor.

It is not sufficient to compare the MSPE for one split. The results below are for 100 iterations of a 70/30 split. For each split we see that the values are close, indicating that the model is as good at predicting in sample as it is out of sample. Note that I am using a for-loop.

```
#create two vectors to contain the mse's for each iteration
in.sample.mse<-rep(NA,100)
out.sample.mse<-rep(NA,100)
#repeat 100 times
for(i in 1:100){
cross.val<-sample(1:nrow(Hourly_Pay),0.7*nrow(Hourly_Pay), replace=FALSE)
training.set<-Hourly_Pay[cross.val,]
test.set<-Hourly_Pay[-cross.val,]
#regression
cv.hourlypay.lm<-lm(log(hourpay)~., data=training.set)
#in sample prediction error
in.sample.error=predict(cv.hourlypay.lm,training.set)-log(training.set$hourpay)
#out of sample prediction error
out.sample.error=predict(cv.hourlypay.lm,test.set)-log(test.set$hourpay)
#in sample mse
in.sample.mse[i]<-sum(in.sample.error^2)/length(in.sample.error)
#out of sample mse
out.sample.mse[i]<-sum(out.sample.error^2)/length(out.sample.error)
}
#take the means of the two
mean(out.sample.mse)
```

```
## [1] 0.2152622
```

```
mean(in.sample.mse)
```

```
## [1] 0.209898
```

Using prediction for model comparison

Let's try a simpler model. Let's get rid of all variables except `age` and `gender`. I don't recommend you do this in practice but for the sake of example let's see what happens. We term the model with all the predictors the "full" model and the model with only `age` and `gender` the "reduced" model. Results for the MSPE for both in and out of sample predictions are shown below:

```
##                      full   reduced
## in sample     0.2098488 0.3044208
## out of sample 0.2155342 0.3074014
```

Q: Interpret the results above.

When doing this in practice, it is useful to investigate whether there are any systematic differences between the predictions of one model vs another. This can shed light on how different predictors are associated with the outcome. In our case there are no systematic differences between the plots/errors between the two models.

An extension of this approach is *K-fold cross-validation* which is very common in Machine Learning. In K-fold cross-validation you split your data into k groups and use (k-1) groups together to predict the kth group for all k groups.

Workshop 7: Prediction and Validation

Learning outcomes

Prediction

Using new plots

Doing cross-validation

Hand out the end of term project.

Packages

```
library(arm)
library(ggplot2)
library(dplyr)
library(tidyr)
library(gridExtra)
```

Prediction

We'll run through the Boston house prices data using the methods we covered in the Lecture.

Data description The Boston data frame has 506 rows and 14 columns.

- CRIM: per capita crime rate by town.
- ZN: proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS: proportion of non-retail business acres per town.
- CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- NOX: nitrogen oxides concentration (parts per 10 million)
- RM: number of rooms per dwelling.
- AGE: proportion of owner-occupied units built prior to 1940 in the town.
- DIS: weighted mean of distances to five Boston employment centres.
- RAD: index of accessibility to radial highways.
- TAX: full-value property-tax rate per $10,000.
- PT: pupil-teacher ratio by town.
- LSTAT: lower status of the population (percent) in the town.
- MV: median value of owner-occupied homes in $1000s.

MV is the outcome variable.

Notes

The "town" in PT can be seen as the local neighbourhood.

Some predictors have many 0 values (e.g. ZN) so it might be worth turning them into categorical or binary variables. Other predictors have many points that are small and then some that are larger. For example you could make CRIM binary by saying if CRIM<5 then CRIM.BIN=0 and otherwise CRIM.BIN=1

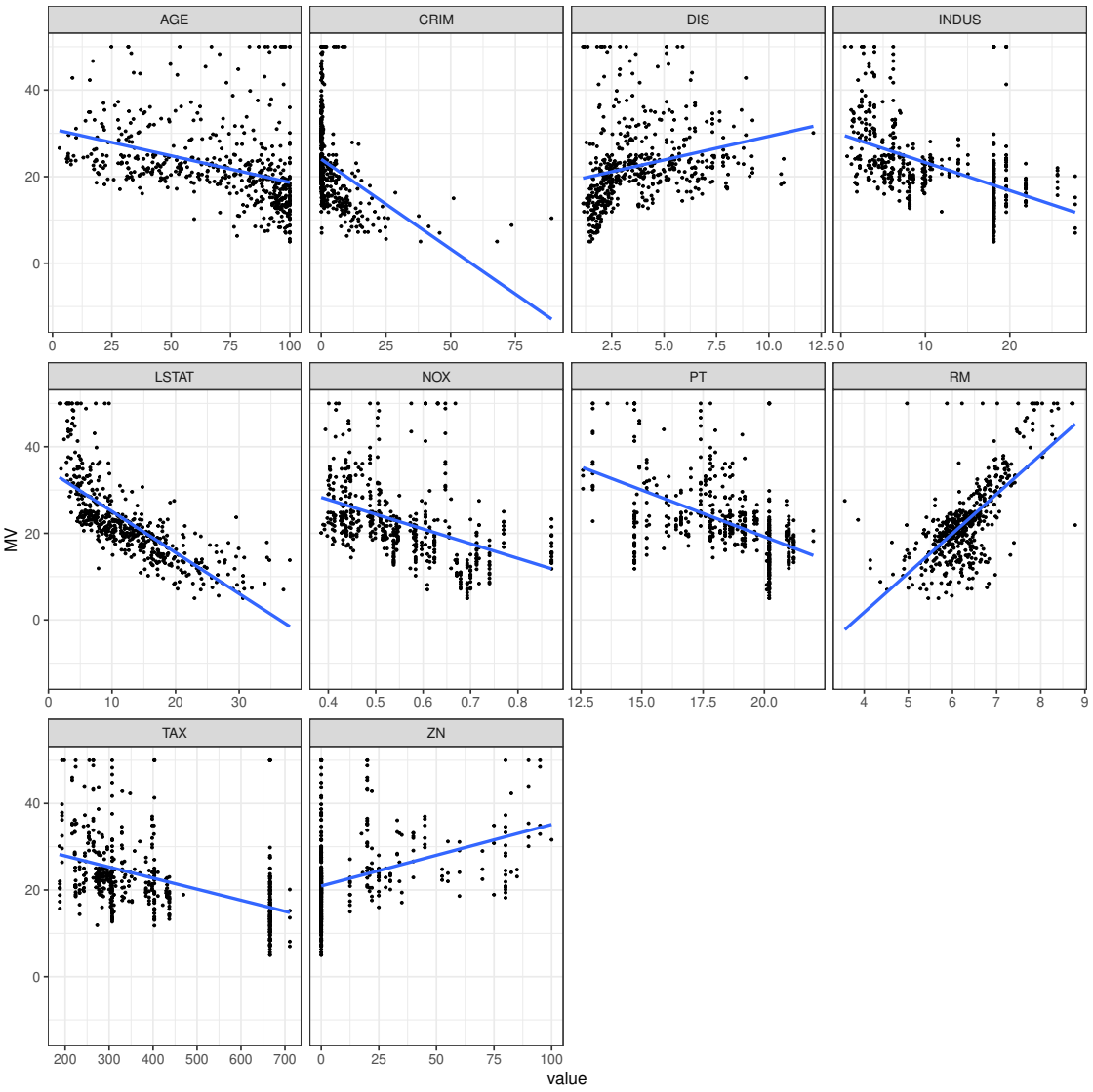
No-one really knows what RAD is so don't worry too much about this variable when it comes to interpretation.

Plotting

The plot below allows you to plot all the predictors against the outcome. It also adds a line of best fit for the univariate regression. Bear in mind that you should probably separate the categorical variables (e.g. RAD and CHAS from the continuous ones)

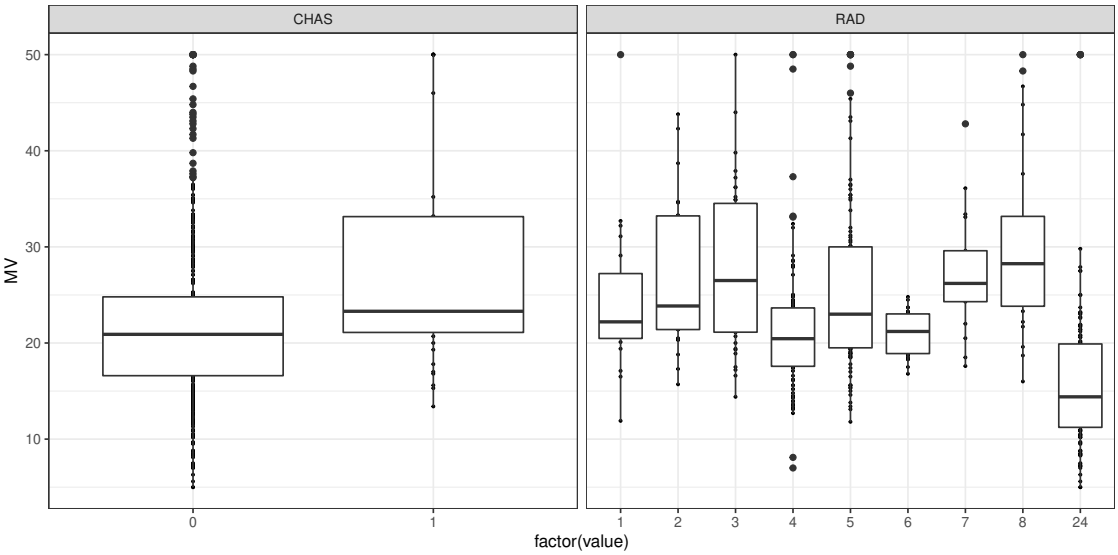
Load the data and create the plots for continuous predictors:

```
Boston.dat <- read.csv("boston.csv", header = TRUE)
special.dat <- gather(data = Boston.dat[, -c(4, 9)], -MV, key = "var", value = "value")
# gather (and similar functions like melt) stack the continuous predictor
# variables on top of each other.  This allows us to use facet_wrap when the
# variables have different attributes We remove columns 2 and 4 as these are
# the categorical variables To see the data type: View(special.dat) now for
# the plot using special dat and scales='free_x' free_x means that the
# x-axis is not the same for each one of the plots
p1 <- ggplot(special.dat, aes(x = value, y = MV)) + geom_point(size = 0.5) +
    geom_smooth(method = "lm", se = FALSE) + facet_wrap(~var, scales = "free_x")
# for book
p1 <- p1 + theme_bw() + scale_fill_grey() + theme(legend.position = "none")
p1
```



Create plots for the categorical predictors:

```
# categorical predictors (only one in this case)
special.dat <- gather(data = Boston.dat[, c(4, 9, 13)], -MV, key = "var", value = "value")
p2 <- ggplot(special.dat, aes(x = factor(value), y = MV)) + geom_point(size = 0.5) +
    geom_boxplot() + facet_wrap(~var, scales = "free_x")
# for book
p2 <- p2 + theme_bw() + scale_fill_grey() + theme(legend.position = "none")
p2
```



Use the `predict` function.

1. The first argument is the name of the linear model you want to predict from.
2. The second is the data frame of the values of the predictor variables you want to predict for. `interval="prediction"` says to use the predictive interval which means that the further away from the centre of the data the wider the interval.

Let's take an existing data point and predict the MV for that point.

```
Boston.lm<-lm(MV~.,data=Boston.dat)
old.point<-Boston.dat[10,]
predict(Boston.lm,old.point,interval="predict", level=0.95)
```

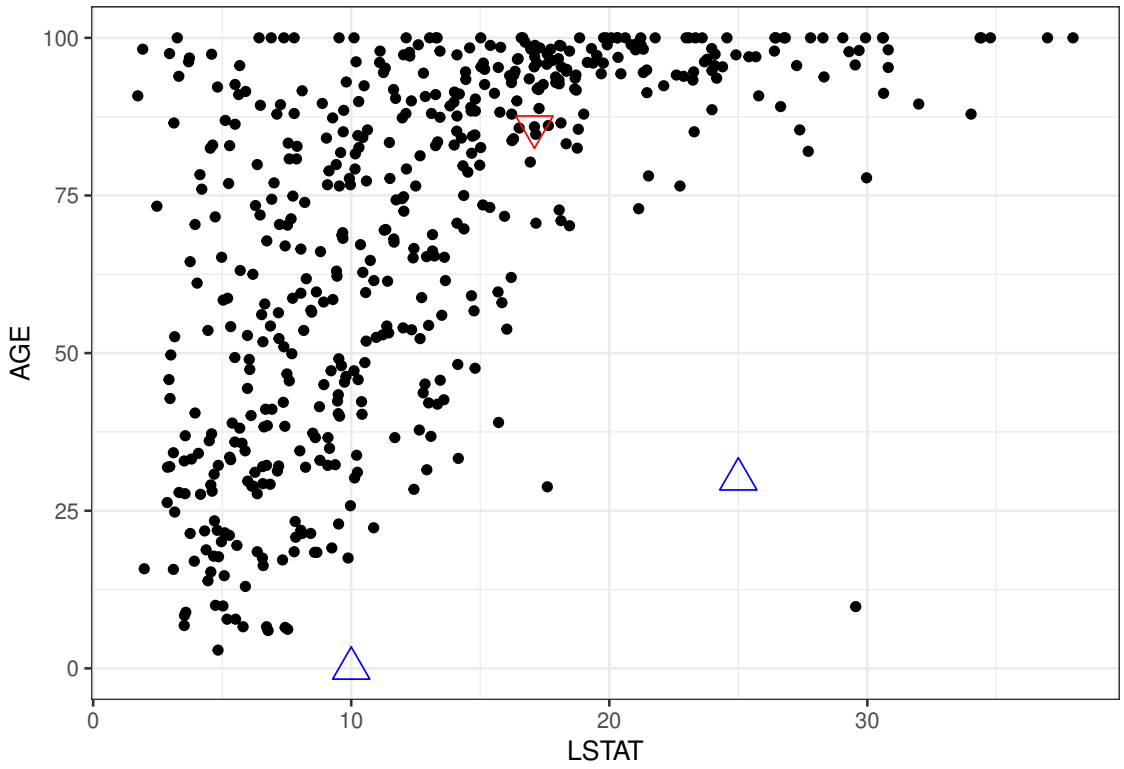
```
##         fit      lwr      upr
## 10 18.71967 9.157267 28.28207
```

This is an example of *in sample* prediction as we are taking an existing observation. Another example of in sample prediction would be if we used a *new* observation but where all the values of the predictors were within the observed ranges for the predictor *and* within the general trend of the data for the predictors.

For example if we look at a plot of two predictors against one another LSTAT vs AGE and consider the point in the middle of the red triangle, it is an in-sample point whereas the two points represented by the blue triangles are out of sample points even though they are both within the ranges of AGE and LSTAT.

```
p1 <- ggplot(Boston.dat, aes(x = LSTAT, y = AGE)) + geom_point()
# highlight the point
p1 <- p1 + geom_point(data = old.point, shape = 6, colour = "red", size = 5) +
    theme_bw()
# out-of-sample points
new.points <- data.frame(LSTAT = c(10, 25), AGE = c(0, 30))
```

```
p1 <- p1 + geom_point(data = new.points, shape = 2, colour = "blue", size = 5)
p1
```



Cross-validation

The basic idea of cross-validation is as follows:

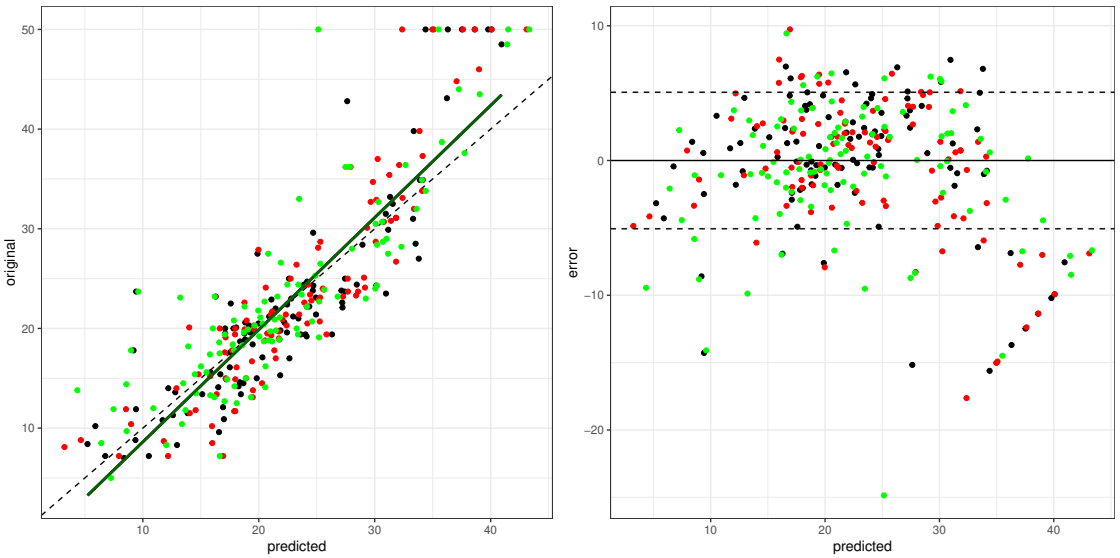
1. Take a random % (e.g. 90%) of the data. These are called the *training set*. The remaining % (e.g. 10%) are called the *test set*.
2. Fit a model using the training set data only.
3. Using the model and the values of the predictors for the test set predict the outcome for the test set. These are your predicted outcomes.
4. Finally, compare the predicted outcomes to the observed outcome in the test set.

Do this a few times with different 90/10 splits and see which models do best. You can also try an 80/20 or 70/30 split.

Let's apply cross-validation to the Boston housing data. We'll create 3 80/20 splits: In order to do cross-validation we need a random sample of 80%: `0.8*nrow(Boston.dat)` .

```
for(i in 1:3){
#create training/test sets
cross.val<-sample(1:nrow(Boston.dat),0.8*nrow(Boston.dat) , replace=FALSE)
training.set<-Boston.dat[cross.val,] #the 80% to fit the model
test.set<-Boston.dat[-cross.val,] # the 20% to use as validation sample
#fit the model
cv.MV.lm<-lm(MV~., data=training.set)
#create data frame to use in plots
pred.val.set<-data.frame(predicted=predict(cv.MV.lm,test.set),
#predicted vs original
original=test.set$MV,error=(predict(cv.MV.lm,test.set)-test.set$MV))
```

```
#first iteration
if(i==1){
p1<-ggplot(data=pred.val.set, aes(x=predicted,y=original))+geom_point()+theme_bw()
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE)
#the ideal would be for the lm to fit the diagonal
p1<-p1+geom_abline(slope=1,intercept=0, linetype="dashed")
#predicted vs error
p2<-ggplot(data=pred.val.set, aes(x=predicted,y=error))+geom_point()+theme_bw()
}else{

  #points for the second iteration
  if(i==2){
    #points on LHS plot
p1<-p1+geom_point(data=pred.val.set, aes(x=predicted,y=original), color="red")
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE, color="darkred")

#points on RHS plot
p2<-p2+geom_point(data=pred.val.set, aes(x=predicted,y=error), color="red")
}else{

#points for the third iteration
#points on LHS plot
p1<-p1+geom_point(data=pred.val.set, aes(x=predicted,y=original), color="green")
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE, color="darkgreen")

#points on RHS plot
p2<-p2+geom_point(data=pred.val.set, aes(x=predicted,y=error), color="green")
#lines at 0 and +/- one std deviation of error
p2<-p2+geom_abline(slope=0,intercept=sd(pred.val.set$error), linetype="dashed")
p2<-p2+geom_abline(slope=0,intercept=0)
p2<-p2+geom_abline(slope=0,intercept=-sd(pred.val.set$error), linetype="dashed")
}}}
grid.arrange(p1,p2,nrow=1)
```



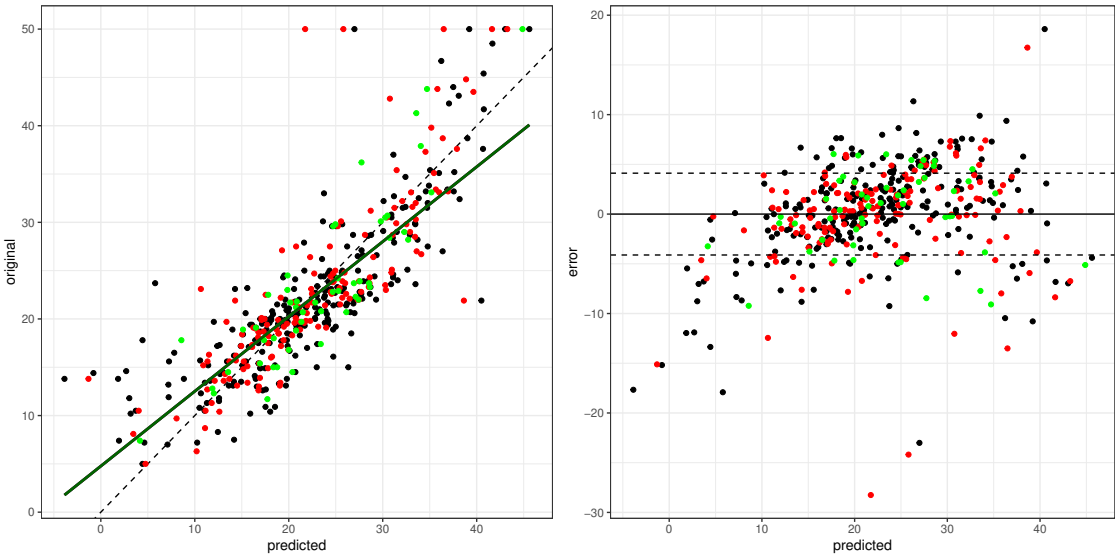
Q: Comment on the results above. For the predicted vs original plot: Consider the x,y spreads. Compare the regression line with the dashed diagonal. For the predicted vs error plot: Is there potentially a trend in this plot or are the data a random scatter? Are the predictions too variable for the model to be good for prediction? For both plots: Is there a significant difference between the 3 iterations?

Using different splits

Let's see what the effect is of having different splits. We'll try 50/50, 70/30 and 90/10 splits. I don't recommend the first one, but it is interesting to see what happens.

```
split.proportions<-c(0.5,0.7,0.9)
for(i in 1:3){
#create training/test sets
cross.val<-sample(1:nrow(Boston.dat),split.proportions[i]*nrow(Boston.dat) , replace=FALSE)
training.set<-Boston.dat[cross.val,] #the 50/70/90% to fit the model
test.set<-Boston.dat[-cross.val,] # the 50/30/10% to use as validation sample
#fit the model
cv.MV.lm<-lm(MV~., data=training.set)
#create data frame to use in plots
pred.val.set<-data.frame(predicted=predict(cv.MV.lm,test.set),
#predicted vs original
original=test.set$MV,error=(predict(cv.MV.lm,test.set)-test.set$MV))
#first iteration
if(i==1){
p1<-ggplot(data=pred.val.set, aes(x=predicted,y=original))+geom_point()+theme_bw()
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE)
#the ideal would be for the lm to fit the diagonal
p1<-p1+geom_abline(slope=1,intercept=0, linetype="dashed")
#predicted vs error
p2<-ggplot(data=pred.val.set, aes(x=predicted,y=error))+geom_point()+theme_bw()
}else{
#points for the second iteration
  if(i==2){
```

```
    #points on LHS plot
p1<-p1+geom_point(data=pred.val.set, aes(x=predicted,y=original), color="red")
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE, color="darkred")

#points on RHS plot
p2<-p2+geom_point(data=pred.val.set, aes(x=predicted,y=error), color="red")
}else{
#points for the third iteration
#points on LHS plot
p1<-p1+geom_point(data=pred.val.set, aes(x=predicted,y=original), color="green")
#regress one on the other to see "fit"
p1<-p1+geom_smooth(method="lm", se=FALSE, color="darkgreen")

#points on RHS plot
p2<-p2+geom_point(data=pred.val.set, aes(x=predicted,y=error), color="green")
#lines at 0 and +/- one std deviation of error
p2<-p2+geom_abline(slope=0,intercept=sd(pred.val.set$error), linetype="dashed")
p2<-p2+geom_abline(slope=0,intercept=0)
p2<-p2+geom_abline(slope=0,intercept=-sd(pred.val.set$error), linetype="dashed")
}}}
grid.arrange(p1,p2,nrow=1)
```



We can see that there is a bit less scatter for the green/light (90/10) than the red/middle (70/30) than the black (50/50) points. The 90/10 split can be slightly deceptive because it sometimes appears that the predictions are very good. This is why it is important to run the same split a number of times.

Mean square predictive errors

Let's see how the 80/20 split does in terms of MSPE

```
#create two vectors to contain the mse's for each iteration
in.sample.mse<-rep(NA,100)
out.sample.mse<-rep(NA,100)
#repeat 100 times
for(i in 1:100){
cross.val<-sample(1:nrow(Boston.dat),0.8*nrow(Boston.dat), replace=FALSE)
```

```
training.set<-Boston.dat[cross.val,]
test.set<-Boston.dat[-cross.val,]
#regression
cv.MV.lm<-lm(MV~., data=training.set)
#in sample prediction error
in.sample.error=predict(cv.MV.lm,training.set)-training.set$MV
#out of sample prediction error
out.sample.error=predict(cv.MV.lm,test.set)-test.set$MV
#in sample mse
in.sample.mse[i]<-sum(in.sample.error^2)/length(in.sample.error)
#out of sample mse
out.sample.mse[i]<-sum(out.sample.error^2)/length(out.sample.error)
}
#take the means of the two
mean(out.sample.mse)
```

```
## [1] 24.51077
```

```
mean(in.sample.mse)
```

```
## [1] 22.20993
```

The two are close with a difference of between 2 and 3.

Using prediction for model comparison

Let's get rid of all variables except `RM` and `LSTAT` and run a 70/30 split.

```
##                    full  reduced
## in sample     22.28763 30.56353
## out of sample 24.01834 30.91638
```

If we compare the full to the reduced model MSPE we see that the reduced model MSPE is substantially higher than that of the full model. This indicates that the full model is _____ at predicting the outcome than the reduced model. If we look at the difference between the in sample and out of sample MSPE for the full model we see that this is similar to that for the 80/20 split and is between 2 and 3. The difference in MSPE for the the reduced model is somewhat smaller between 1 and 2 meaning that the reduced model is a bit better at predicting the out of sample points than the full model.

Lecture 8: Introduction to Logistic Regression

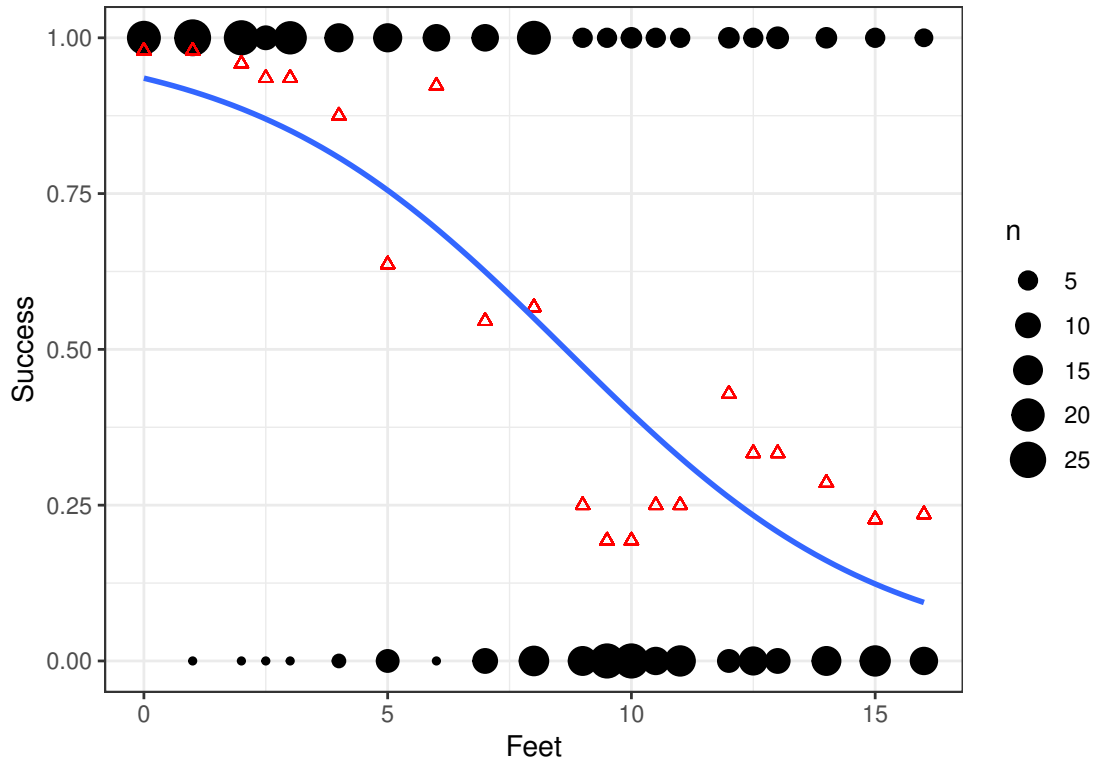
Learning outcomes

Understanding of the logistic function and how it is used to link a continuous predictor with a binary outcome

Interpreting the coefficients in the case of a single continuous predictor

Plot of observed probabilities vs predictor

The plot below shows the probability of successfully putting a golf ball against the distance from the hole (feet) for a sample of 440 putts. Think about how you could model this relationship.



1. The triangles are:

2. The line is:

3. The black dots are:

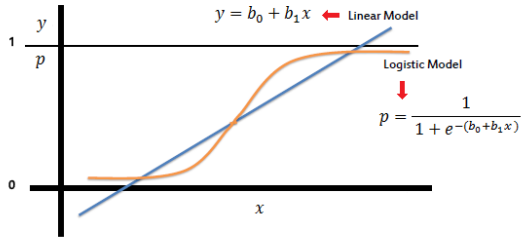
Consider the following:

- Is the relationship linear?
- What problems might you encounter with a linear model in this context
- How might these data have been collected? What would the data in it's raw form look like?
- What is likely success rate for 0 feet? For ∞ feet?

Logistic Regression

A model that solves some of the issues above is the *logistic model*. It maps the real line to a probability rather than a value on the real line or a 0,1.

Generally a logistic regression looks like this (our plot above is reversed because the probabilty decreases with the distance but otherwise the same principles apply)



If $p(y_i = 1)$ is the probability of success and we are considering a single predictor x, then the logistic model says

$$\begin{aligned} p_i &= p(y_i = 1) \\ \log\left(\frac{p_i}{1 - p_i}\right) &= \beta_0 + \beta_1 x_i \\ \text{logit}(p_i) &= \beta_0 + \beta_1 x_i \end{aligned}$$

where

$$\text{logit}(z) = \log\left(\frac{z}{1 - z}\right)$$

We assume that the y_i are independent given the probabilities. The $\beta_0 + \beta_1 x_i$ is linear in the predictors. An equivalent expression which is often used is

$$\begin{aligned} p(y_i = 1) &= \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}} \\ &= \text{logit}^{-1}(\beta_0 + \beta_1 x_i) \\ &= \text{invlogit}(\beta_0 + \beta_1 x_i) \end{aligned}$$

The inverse logit function is non-linear. If we want to calulate p from the equation $\text{logit}(p_i) = \beta_0 + \beta_1 x_i$ we use the following:

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$

This means that a *fixed increase* (e.g. of 1) in a *predictor* does *not* necessarily *result* in a *fixed increase* in the *outcome*. The steepest changes occur in the middle of the curve. This means that logistic regression coefficients can be tricky to interpret. We'll go over many examples in the workshop and quizzes.

The formulae above have one predictor. For multiple predictors simply replace $\beta_0 + \beta_1 x$ with $\beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k$

GLMs and MLE

Logistic regressions are in the family of *Generalised Linear Models*. That is models with a linear fit but a non-linear outcome. The linear part is connected to the non-linear part by a *link* function. For logistic regression this is the logit link. In the last lecture we'll have a quick look at Poisson regression which links a linear part to a model for *count* data using a log link.

There is no closed form solution for the parameters that maximise the logistic regression equivalent to the sum of squared residuals. As a consequence for generalised linear models *Maximum Likelihood Estimation* (MLE) is used to estimate the coefficients of the predictors. In simple terms MLE finds the parameters that maximise the likelihood (probability distribution function) for the model under consideration given the observed data. MLE typically uses numerical optimisation routines although applying MLE to linear regression is equivalent to least squares estimation.

One predictor

As with any data set before you fit a regression it is important to consider what you anticipate the direction of the relationships between the predictors and the outcome to be.

Let's look at the golfing data and fit a line using R.

```
golf.glm<-glm(Success~Feet,data=golf.dat, family=binomial(link="logit"))
#Same as lm but glm and add family="binomial"
display(golf.glm)
```

```
## glm(formula = Success ~ Feet, family = binomial(link = "logit"),
##     data = golf.dat)
##             coef.est coef.se
## (Intercept)  2.67     0.29
## Feet        -0.31     0.03
## ---
##   n = 440, k = 2
##   residual deviance = 460.6, null deviance = 608.9 (difference = 148.3)
```

Interpreting the logistic model

Using the regression output above write down the model for the logistic regression.

The intercept

The intercept is the value of the logit(p(Success)) for 0 Feet: The invlogit of the intercept is the estimated probability of successfully putting a golf ball from 0 Feet.

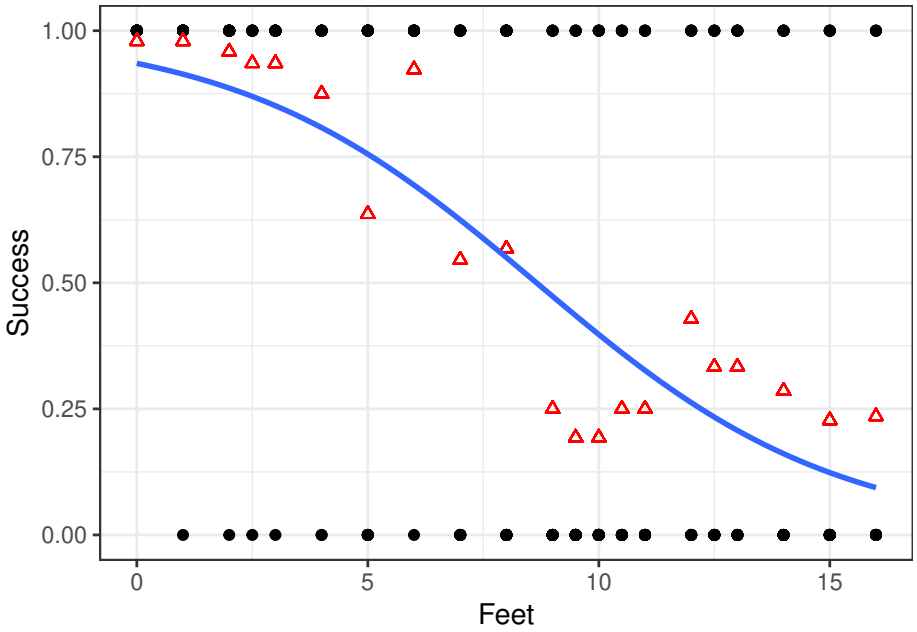
```
invlogit(coef(golf.glm)[1])
```

```
## (Intercept)
##   0.9352132
```

What does the intercept mean in this case?

The slope

Because the model is non-linear we need to choose where to evaluate the predicted outcome. See the plot below:



Q: What is effect on success of moving from 1 foot to 2 feet away?
* Estimate the p(success|1 foot)
* Estimate the p(success|2 feet)
* Calculate their ratio: this tells us something about the relative change in probability – this is called the *risk ratio*

The values you should get are:

```
## (Intercept) (Intercept) (Intercept)
##   0.9138132   0.8862047   0.9697876
```

So the probability of success at 1 foot is 91.6% and at 2 feet it is 88.8% that's a decrease in 3% of success when moving from 1 to 2 feet away. Not that much.

Now let's do the same thing but in the centre of the curve where it is steepest.

Q: What is effect on success of moving from 7 foot to 8 feet away?
* Estimate the p(success|7 feet)
* Estimate the p(success|8 feet)
* Calculate their ratio

You should get that there is a change in 12% decrease in successful putting when you move from 7 to 8 feet.

Interpreting coefficients of predictors as odds ratios

Consider the case where Feet=2. The probability of success is then $p($ Success $\mid$ Feet $= 2) = 0.89$. The *odds* of Success are $\frac{p}{1-p}$ which is $\frac{0.89}{1-0.89} = 8.09$. So the odds of success are high at 2 feet.

If we consider two cases, where Feet=2 and Feet =1 we get two odds. Their ratio – the *odds ratio* – is another meausure of relative change in Success:

- Feet 2 odds: $\frac{0.8894}{1-0.8894} = 8.0416$
- Feet 1 odds: $\frac{0.9159}{1-0.9159} = 10.8906$
- Odds ratio: $\frac{8.041}{10.806} = 0.74$
- Coefficient of Feet: $e^{-0.30} = 0.74$

You can check that this holds for any change in 1 in Feet (e.g. from 7-8 feet)

- **Advantages**: The odds ratio doesn't depend on the value of the predictor.
- You can think of the odds ratio as saying something about how success decreases as Feet increase. This is because the odds ratio < 1
- **Disadvantage**: It is hard to interpret. What does an odds ratio actually mean? Unless you're a betting person it's not straightforward.

I may ask you to calulate odds ratios in exams but I don't expect you to try and interpret it other than to comment on the direction of the effect: If the odds ratio is above 1 then the effect is positive,

if it is below 1 the effect is negative. You should find that the direction corresponds to the direction of the risk ratios associated with the same predictor – in the steepest part of the curve. The odds ratio CANNOT be negative.

Workshop 8: Introduction to Logistic regression

Learning outcomes

Running logistic regressions with one predictor in `R`

Logistic Regression

Logistic regression is the standard way to model *binary* outcomes, data where the *outcome* is in the form 0/1 (or "yes, no" etc.)

We'll look at data on survival of 237 patients admitted to an intensive care unit (ICU) in an American Hospital.

```
icu.dat<-read.csv("ICUdeaths.csv", header=TRUE)
head(icu.dat)
summary(icu.dat)
```

The data are as follows:

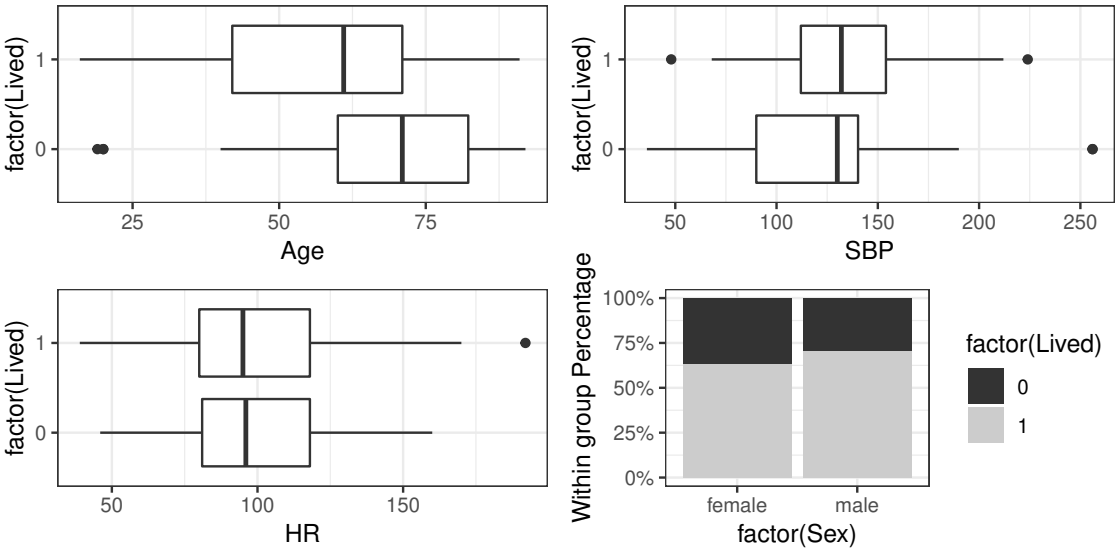
- `Lived`: lived to discharge (=1) , died in ICU (=0)
- `Age`: Age of patient in Years
- `Sex`: female, male
- `SBP`: Systolic blood pressure at admission – higher is bad for you
- `HR`: heart rate at admission – higher is bad for you

`Age`,`SBP` and `HR` are continuous predictors.

Let's look at the data. Because the outcome is binary, the best way to visualise the relationships between the outcome and the continuous predictors is via a boxplot. Bear in mind that the logistic model that relates the outcome to the continuous predictors is not linear so some caution is required when interpreting the boxplots. To remind you of this, I have flipped them on their side.

Finally to model the relationship between categorical predictors we use bar plots.

```
p1 <- ggplot(icu.dat, aes(x = factor(Lived), y = Age)) + geom_boxplot() + coord_flip()
p1 <- p1 + theme_bw() + theme(legend.position = "none") + scale_fill_grey()

p2 <- ggplot(icu.dat, aes(x = factor(Lived), y = SBP)) + geom_boxplot() + coord_flip()
p2 <- p2 + theme_bw() + theme(legend.position = "none") + scale_fill_grey()

p3 <- ggplot(icu.dat, aes(x = factor(Lived), y = HR)) + geom_boxplot() + coord_flip()
p3 <- p3 + theme_bw() + theme(legend.position = "none") + scale_fill_grey()

# cross tabulation plot: are two categorical variables related?
p4 <- ggplot(icu.dat, aes(x = factor(Sex), fill = factor(Lived))) + geom_bar(position = "fil
    scale_y_continuous(name = "Within group Percentage", labels = scales::percent)
p4 <- p4 + theme_bw() + scale_fill_grey()

grid.arrange(p1, p2, p3, p4, nrow = 2)
```



Let's focus on the continuous predictors for now. We'll return to the binary predictors later in the workshop.

Q: Based on the plots above, which of the three *continuous* predictors do you expect to be associated with Lived?

Specifically for Age, what do you expect the relationship between ICU survival and Age to be? Will the coefficient of Age be positive or negative? Will the odds ratio associated with this coefficient be above or below 1?

One continuous predictor

Let's start by adding Age. The `R` code function for logistic regression is similar to that for a linear regression. There are two important differences in the *name* of the function which is `glm` for *Generalised linear model*. `glm` also has an additional *argument*: `family` which takes on the value `binomial(link="logit"))`. This is because there are other potential generalised linear models such as Poisson or negative binomial.

```
icu.glm<-glm(Lived~Age,data=icu.dat,family=binomial(link="logit"))
display(icu.glm)
```

```
## glm(formula = Lived ~ Age, family = binomial(link = "logit"),
##     data = icu.dat)
##             coef.est coef.se
## (Intercept)  3.19     0.60
## Age         -0.04     0.01
## ---
##   n = 237, k = 2
##   residual deviance = 273.4, null deviance = 297.4 (difference = 23.9)
```

Write down the equation for the fitted model

Interpreting the coefficients

In order to get an idea of what the coefficients mean let's look at the plot.

Plotting the model

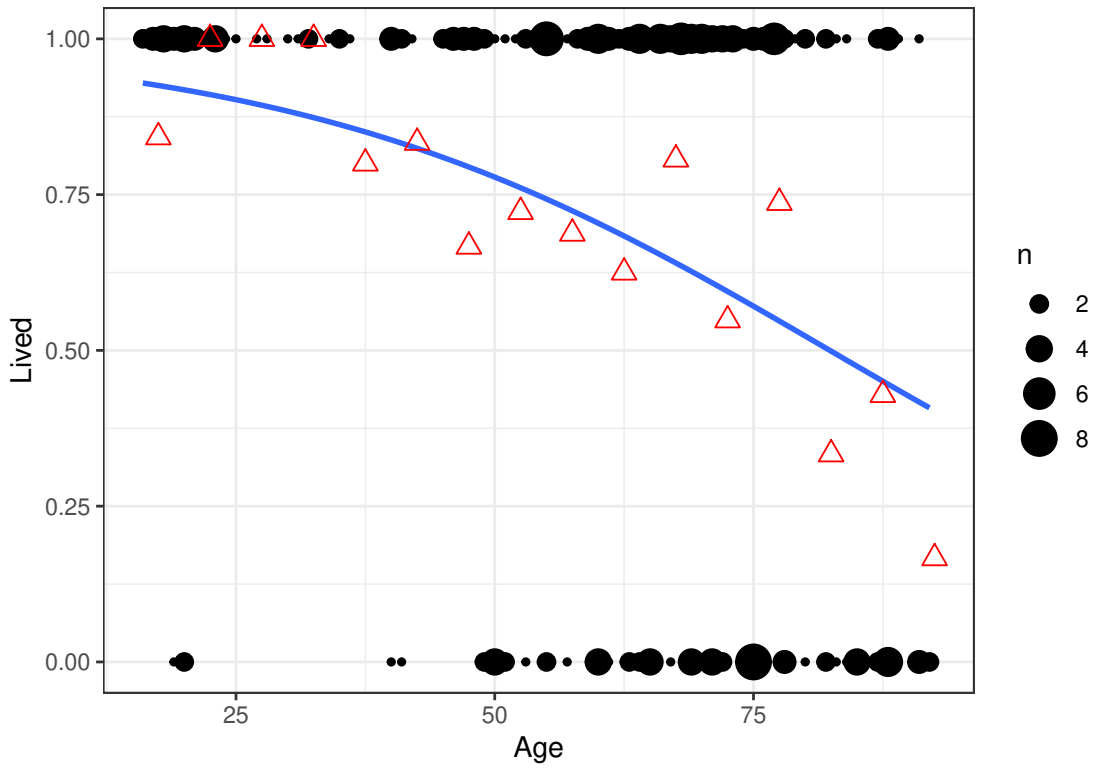
First we need to prepare the data. The plot below shows on the x-axis the `Age` in the data grouped into age bands of 5. On the y-axis are the observed proportions of Lived (number Lived/total) in each `Age` band. Therefore we need to

- create `Age` bands (called) "bins",
- estimate the proportions of patients who `Lived=1` in each `Age` band.

I'll go over the code carefully as it is a bit complex. The difficult bit is getting the proportion of people who have lived in each age bands. We do this first:

```
#create a new data frame (icu.breaks)
#which adds a column to icu.dat using the mutate function in "dplyr".
#It has bins of 5 years starting at 15 and ending at 95
icu.breaks<-mutate(icu.dat, bin=cut(Age,breaks = seq(15, 95, by=5)))

#estimate the proportion of people that survived the ICU in each bin (y-axis)
prop <- prop.table(with(icu.breaks, table(Lived,bin)),2)[2,]
# the midpoint of each bin (x-axis)
midbin<- seq(17.5, 92.5, by=5)
#create a new data frame using these two variables
icu.bin<-data.frame(midbin,prop)
#icu.bin
```

The graph below is saying that for people of `Age` 15 about 90% survive once admitted to the ICU. This decreases to around 40% for 95 year old. Based on the points on `Lived`=0 and 1 we see that people of all ages survive but that older people are much more likely to die (with very few die below the age of 50).



Intercept

The intercept is not really relevant in this example. The reason is that very small children (under 2) have a different pattern of survival when admitted to ICU than adults or older children. As with linear regression it is possible to centre and or standardise predictors in order to better interpret or compare the relative strength of coefficients.

Below is the output for the regression with centred `Age`.

```
cent.Age<-with(icu.dat, (Age-mean(Age)))
cent.age.glm<-glm(Lived~cent.Age, data=icu.dat, family = binomial(link="logit"))
display(cent.age.glm)
```

```
## glm(formula = Lived ~ cent.Age, family = binomial(link = "logit"),
##     data = icu.dat)
##             coef.est coef.se
## (Intercept)  0.87     0.15
## cent.Age    -0.04     0.01
## ---
##   n = 237, k = 2
##   residual deviance = 273.4, null deviance = 297.4 (difference = 23.9)
```

Interpret the intercept of the model with the centred "cent.Age" replacing "Age".

Coefficient of Age

The model above is almost linear over the range of Age. This means that if we calculate the risk ratio associated with a change in 10 years at around Age 20-29 and then 70-79 the relative change will be similar for the whole range. Use the non-centered regression to do this as you don't then need to adjust the values of Age.

Calculate the risk ratio associated with change from "Age" 20-29 and then another risk ratio for 70-79. Are these very different?

Predicted vs Observed table – a.k.a classification table

One way to see how good the logistic model is to compare the observed outcome to the predicted outcome – similar to validation. As a logistic regression outputs probabilities as opposed to a 0/1 outcome, we make the simple assumption that if a predicted probability exceeds 0.5 this corresponds to a predicted outcome=1

```
pred.glm<-as.numeric(icu.glm$fitted.values>0.5)
glm.dat<-data.frame(predicted=pred.glm, observed=icu.dat$Lived)
table(glm.dat)
```

```
##           observed
## predicted   0   1
##         0  19   9
##         1  57 152
```

We can see that overall the model predicts correctly $\frac{19+152}{237} = 0.72$, i.e. 72% of the time. For those who lived $\frac{152}{152+57} = 0.73$ i.e. 73% are correctly predicted and for those who died $\frac{19}{19+9} = 0.68$ i.e 68% are correctly predicted. On all counts the model is doing well in terms of prediction.

A single binary predictor

Let us now consider a single binary predictor in a logistic regression. When the outcome and the predictor are continuous we look at a scatterplot with the outcome on the y-axis and the predictor on the x-axis. When the outcome is continuous and the predictor is categorical we look at a scatterplot with the outcome on the y-axis and the predictor on the x-axis. We saw above that with a binary outcome and a continuous predictor we can look at "tilted" boxplots where the predictor is on the x-axis and the binary outcome on the y-axis. How do we visualise a relationship between a binary outcome and a categorical predictor?

First off we can produce a table that tells us how Sex and Lived are related. We can do this by looking at whether the relative proportions of each level of one category are similar across the levels of the other category.

```
round(prop.table(with(icu.dat, table(Lived,Sex)),2),2)
```

```
##      Sex
## Lived female male
##     0   0.36 0.29
##     1   0.64 0.71
```

The table tells us that amongst the females approximately _____% died and _____% survived the ICU and amongst the males approx _____% died and _____% survived the ICU. The proportions are similar but not identical, we might therefore concluded that we would expect at most a weak relationship (although this depends on sample size).

As we saw in the plots at the beginning of the workshop, there is also a graphical representation of this table in figure A below the code. We see that the grey bars are at a similar levels corresponding to 29 and 36 %, indicating that Sex probably does not have a big impact on Lived.

What do a table and plot look like if the predictor *is* associated with the outcome? Some code below where we artificially generate binary `A` and `B`. The code also shows what `A` and `B` look like as well as the table of the relative proportions.

```
#fake data with an association:
#A binary with 30 1's and 50 0's.
A<-c(rep(1,30),rep(0,50))
#rbinom generates n binary random variables with p(x=1)=p.
#size picks a sample of size from these
B<-ifelse(A==1,rbinom(n=100,size=1,p=0.2),rbinom(n=100,size=1,p=0.8))
#show A,B
A
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
## [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [71] 0 0 0 0 0 0 0 0 0 0
```

```
B
```

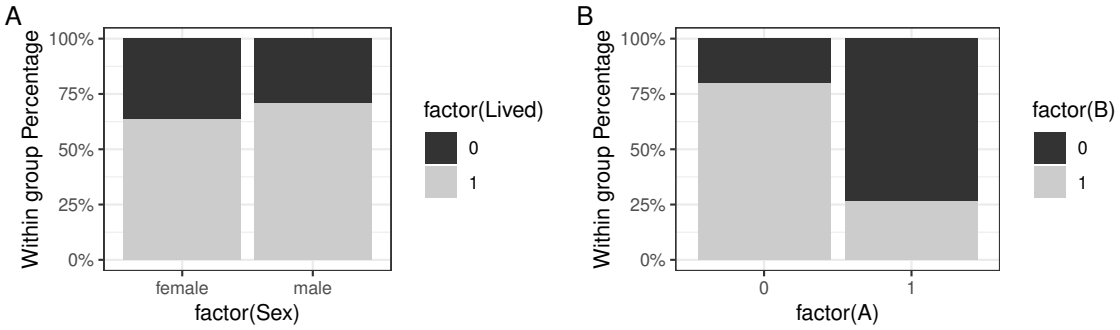
```
##  [1] 0 0 1 1 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 1 1
## [36] 1 1 0 1 1 1 1 1 1 0 1 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 0
## [71] 1 1 1 0 1 1 1 1 1 1
```

```
#There is a strong dependence between A and B
AB<-data.frame(A=A, B=B)
round(prop.table(table(A,B),2),2)
```

```
##    B
## A     0    1
##   0 0.31 0.83
##   1 0.69 0.17
```

Plot B has the corresponding proportion plot for `AB`:

```
p1 <- ggplot(icu.dat, aes(x = factor(Sex), fill = factor(Lived))) + geom_bar(position = "fill") +
    scale_y_continuous(name = "Within group Percentage", labels = scales::percent)
p1 <- p1 + theme_bw() + scale_fill_grey() + labs(tag = "A")

p2 <- ggplot(data = data.frame(A, B), aes(x = factor(A), fill = factor(B))) +
    geom_bar(position = "fill") + scale_y_continuous(name = "Within group Percentage",
    labels = scales::percent)
p2 <- p2 + theme_bw() + scale_fill_grey() + labs(tag = "B")

grid.arrange(p1, p2, nrow = 1)
```



Q: Based on the table and the graphs above, do you think A is an important predictor of B? How do you know? Contrast this to the relationship between `Sex` and `Lived`.

Let's run the univariate regression of `Sex` on `Lived`.

```
icu.glm.bin<-glm(Lived~Sex,data=icu.dat, family=binomial(link="logit"))
display(icu.glm.bin)
```

```
## glm(formula = Lived ~ Sex, family = binomial(link = "logit"),
##     data = icu.dat)
##             coef.est coef.se
## (Intercept) 0.56     0.21
## Sexmale     0.34     0.28
## ---
##   n = 237, k = 2
##   residual deviance = 296.0, null deviance = 297.4 (difference = 1.4)
```

Q: Does the output of display() agree with your comments above?

Lecture 9: More Logistic Regression

Learning outcomes

Exploring the diagnostics for logistic regression

Adding multiple predictors

Understanding Average Predictive Comparisons

Diagnostics for Logistic Regression

As with linear regression, there are diagnostics for logistic regression. Again it is important *not* to focus on these exclusively when performing an analysis but rather use them as part of a tool-kit.

Let's use the golfing data again:

```
golf.dat<-read.csv("golf.full.csv",header = TRUE)
golf.glm<-glm(Success~Feet, data=golf.dat, family=binomial(link="logit"))
```

Statistical Significance of coefficients

The output of `display` for `glm` is similar to that for `lm`. Similarly the coefficients are significant at the 5% level if coefficient $\pm$ 2 $\times$ se does not cover 0.

```
## glm(formula = Success ~ Feet, family = binomial(link = "logit"),
##     data = golf.dat)
##             coef.est coef.se
## (Intercept)  2.69     0.13
## Feet        -0.30     0.01
## ---
##   n = 2200, k = 2
##   residual deviance = 2311.5, null deviance = 3039.3 (difference = 727.8)
```

Q: For the ouput of `display()` above, are the intercept or the coefficient of `Feet` significant? How can you tell?

Classification table - predicitve ability of model

We saw in the last workshop that we can use the table that plots the predicted vs the observed outcomes as a measure of how good the model predicts/fits the data. Let's try this with the golfing data.

First we create a function (details in next week's workshop) that allows us to calculate the percentage of the time our model predicts correctly.

```
# function arguments create the data frame
ct.op <- function(predicted, observed) {
    df.op <- data.frame(predicted = predicted, observed = observed)
    # create a table
    op.tab <- table(df.op)
```

```
    # use the prop.table function to obtain the rows we need and stack them on
    # top of each other with rbind
    op.tab <- rbind(op.tab, c(round(prop.table(op.tab, 2)[1, 1], 2), round((prop.table(op.tab,
        2)[2, 2]), 2)))
    # name the rows
    rownames(op.tab) <- c("pred=0", "pred=1", "%corr")
    # name the columns
    colnames(op.tab) <- c("obs=0", "obs=1")
    # return the table
    op.tab
}
```

Now apply the function to the golf data:

```
##         obs=0  obs=1
## pred=0 792.00 308.00
## pred=1 232.00 868.00
## %corr    0.77   0.74
```

Q: Looking at the classification table, is this a good model for prediction? Explain your answer.

Deviance

Another diagnostic is the model *deviance*. The deviance (also known as the -2 log likelihood) is analogous to the F-statistics in that it says something about the model as a whole but not individual predictors. The higher the deviance, the worse the model fit *relative to the null model* deviance.

The deviance can also be seen as a logistic regression analog of the residual sum of squares – so typically, the bigger it is, the worse the model fit. The deviance is most often used to formally compare *nested* models but can be used to informally compare non-nested models.

Q: What does nested mean?

Deviance in R

The output of `summary()` or `display()` for a glm gives the *null* and the *residual deviance*

- The null deviance: for the *null* model with *only* the intercept (no predictors) - dev_0
- The residual deviance: the deviance for the model estimated by the glm - dev_m

- The difference between the deviances δ_d = dev_0 - dev_m is a measure of how much better the estimated model is than the null model – so the higher it is, the better the model is than the intercept only model
- It is distrubuted according to the χ^2 distribution on k degrees of freedom where k is the number of predictors in the model

How to use the deviance:

- For 1 predictor the critical value at the 5% level is 3.84:
- For 1 degree of freedom if δ_d>3.84 then the estimated model is better than the null model
- For k predictors let c_k be the critical value for the χ^2 distribution on k degrees of freedom
- If the δ_d>c_k then the estimated model is better than the null model
- If the δ_d<c_k then the estimated model is not better than the null model

Let's look at the data from the last workshop on ICU deaths. First consider the model with only `Age`.

```
icu.dat<-read.csv("ICUdeaths.csv", header=TRUE)
icu.glm.1<-glm(Lived~Age,dat=icu.dat, family=binomial(link="logit"))
display(icu.glm.1)
```

```
## glm(formula = Lived ~ Age, family = binomial(link = "logit"),
##     data = icu.dat)
##             coef.est coef.se
## (Intercept)  3.19     0.60
## Age         -0.04     0.01
## ---
##   n = 237, k = 2
##   residual deviance = 273.4, null deviance = 297.4 (difference = 23.9)
```

Q: Look at the output of`display(icu.glm.1)` above. What are the null and residual deviances? What is the difference between them? Based on this information which is better, the null model or `icu.glm.1`?

Adding more predictors

Let's continue using the data on ICU deaths and add more predictors. In this case `SBP` (systolic blood pressure).

```
icu.glm.2<-glm(Lived~Age+SBP,dat=icu.dat, family=binomial)
display(icu.glm.2)
```

```
## glm(formula = Lived ~ Age + SBP, family = binomial, data = icu.dat)
##             coef.est coef.se
## (Intercept)  1.70     0.81
## Age         -0.04     0.01
## SBP          0.01     0.00
## ---
```

```
##   n = 237, k = 3
##   residual deviance = 265.8, null deviance = 297.4 (difference = 31.6)
```

Deviance again

Note that SBP is significant at the 5% level with `coef.se=0.004`. The model `icu.glm.2` is better than model `icu.glm.1` as it has an additional significant predictor. However we can confirm this by using the residual deviances to compare them to one another – because `icu.glm.1` is nested in `icu.glm.2`.

- The redisual deviance for the model `icu.glm.1` - dev_1 =_____
- The residual deviance for the model `icu.glm.2` - dev_2 =_____
- The difference between the deviances δ_d = dev_2 - dev_1 =_____ is a measure of how much better `icu.glm.2` is than `icu.glm.1`
- It is distrubuted according to the χ^2 distribution on 1 degrees of freedom because there are _____ predictors in `icu.glm.2` and _____ predictors `icu.glm.1`
- Is δ_d>3.84? _____
- Therefore `icu.glm.2` is _____ than `icu.glm.1`

Let's run a model with `Age` and `Sex` as predictors:

```
icu.glm.3<-glm(Lived~Age+Sex,dat=icu.dat, family=binomial)
display(icu.glm.3)
```

```
## glm(formula = Lived ~ Age + Sex, family = binomial, data = icu.dat)
##             coef.est coef.se
## (Intercept)  3.07     0.65
## Age         -0.04     0.01
## Sexmale      0.14     0.30
## ---
##   n = 237, k = 3
##   residual deviance = 273.2, null deviance = 297.4 (difference = 24.2)
```

Q: Look at the output `icu.glm.3` above. Is model `icu.glm.1` nested in `icu.glm.3`? If so compare the models using the deviance technique shown above. Which model do you prefer?

More generally if you want to compare two nested models then you obtain the residual deviances, their difference and then compare this to the critical value of the χ^2 on degrees of freedom that are the difference in the number of predictors in the two models. For example, let's run a model with `Age`, `Sex` ,`HBP` and `HR` as predictors:

```
icu.glm.4<-glm(Lived~Age+Sex+SBP+HR,dat=icu.dat, family=binomial)
display(icu.glm.4)
```

```
## glm(formula = Lived ~ Age + Sex + SBP + HR, family = binomial,
```

```
##     data = icu.dat)
##             coef.est coef.se
## (Intercept)  1.20     1.06
## Age         -0.04     0.01
## Sexmale      0.22     0.31
## SBP          0.01     0.00
## HR           0.00     0.01
## ---
##   n = 237, k = 5
##   residual deviance = 265.1, null deviance = 297.4 (difference = 32.3)
```

Q: Is model `icu.glm.3` nested in `icu.glm.4`? If so compare the models using the deviance technique shown above. The critical value for the χ^2 on 3 degrees of freedom is 7.81. Which model do you prefer?

Hypothesis tests:

The when we compare the difference in the deviance with a critical value, we are performing a hypothesis test. Let us consider two logistic regression models:

Model 1: $\text{logit}(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$

Model 2: $\text{logit}(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \ldots + \beta_{k+l} x_{k+l}$

where $l \geq 1$.

When we compare the null and the residual deviance for Model 1 we are running the following test:

$\mathbf{H}_0$: $\beta_1 = \beta_2 = \ldots = \beta_k = 0$

$\mathbf{H}_1$: At least one $\beta_i \neq 0$ where $i \in \{1, 2, \ldots, k\}$

This is equivalent to the F-test for multiple linear regression.

When we compare the residual deviance of Model 1 to that of Model 2 we are running the following test:

$\mathbf{H}_0$: $\beta_{k+1} = \beta_{k+2} = \ldots = \beta_{k+l} = 0$

$\mathbf{H}_1$: At least one $\beta_i \neq 0$ where $i \in \{k+1, k+2, \ldots, k+l\}$

This is like partial F-tests in multiple linear regression.

Using `anova()`

We can use `anova()` (not Anova as we've used before) to run the tests for us:

```r
#compares model with Age and that with Age+Sex
anova(icu.glm.1, icu.glm.3, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: Lived ~ Age
## Model 2: Lived ~ Age + Sex
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       235     273.43
## 2       234     273.21  1  0.22062   0.6386
```

```r
#compares model with Age+Sex and that with Age+Sex+SBP+HR
anova(icu.glm.3, icu.glm.4, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: Lived ~ Age + Sex
## Model 2: Lived ~ Age + Sex + SBP + HR
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       234     273.21
## 2       232     265.09  2   8.1175  0.01727 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Q: Based in the output of the anova()'s above, which model do you prefer? Explain your answer.

Average predictive comparisons

The logistic model is non-linear and that means that the regression coefficients don't have a straightforward interpretation like the coefficients in a linear regression. That means that the effect on the probability of the outcome associated with an increase in one in a predictor depends on the starting value of the predictor and is not the same across the range of the predictor.

If we only have one or two continous predictors we would probably use the plots from the last workshop to get an idea of the curvature of the model and then choose values of the predictors to explore their effect on the outcome. We have 3 continuous predictors so we'll look at *average predictive comparisons* which are a way of obtaining a number that behaves in a similar way to linear regression coefficients.

Maths of APCs

Lecture 10: Poisson models

Learning Outcomes

Basic understanding of Poisson count models

Poisson models

Poisson models are another type of Generalised Linear Model (GLM) where the outcome of interest is not a continuous variable as in linear regression or a binary outcome as in logistic regression but count data. Each point y_i is $\in \{0, 1, 2, \ldots\}$. For example, the number (counts) of Facebook visits per day for a group of people. The function that links the counts with the linear combination of predictors is the *logarithmic link*.

Formally the Poisson distribution is used to model the variation in count data.

$$yi \sim \text{Poisson}(\theta_i)$$
$$\theta_i = \exp(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p)$$

Let's dive straight in: We'll consider the data on Facebook visits.

```
facebook.dat<-read.csv("Facebook.csv",header=TRUE)
head(facebook.dat)
```

```
##   age student gender visits
## 1  21     Yes Female      0
## 2  32      No Female      0
## 3  36      No Female      0
## 4  39      No Female      0
## 5  42     Yes Female      0
## 6  47      No Female      0
```

Let's immediately run a regression:

```
#note that the call to glm has family=poisson rather than family=binomial
facebook.pois<-glm(visits~age+student+gender,data=facebook.dat, family=poisson)
display(facebook.pois)
```

```
## glm(formula = visits ~ age + student + gender, family = poisson,
##     data = facebook.dat)
##             coef.est coef.se
## (Intercept)  2.61     0.11
## age         -0.02     0.00
## studentYes   0.14     0.07
## genderMale   0.02     0.04
## ---
##   n = 308, k = 4
##   residual deviance = 3829.7, null deviance = 4033.1 (difference = 203.3)
```

Q: Write down the model for this regression.

Interpreting the regression coefficients

- The intercept does not mean anything in this context because newborns do not use Facebook. So exp(2.61)=13.6 is the expected number of visits to Facebook of a newborn Female who is not a student. You could centre `age` to interpret this coefficient.
- The coefficient of `age` is the expected difference in log(visits) for an increase in one year of `age`. If we exponentiate the coefficient of `age`. exp(-0.02)=0.98. This means that an increase in one year in age results in a decrease in number of visits to Facebook of 2%.
- The coefficient of StudentYes is the expected difference in log(visits) for students relative to non-students. exp(0.14)=1.15 meaning that the number of visits increasese by 15% for students relative to non-students.
- The coefficient of genderMale is the the expected difference in log(visits) for a males relative to females. exp(0.02)=1.02 meaning that males visit Facebook 2% more often than females.

Diagnostics

These are the same as those for the logistic models and are intepreted in the same way.

Other things to consider when using Poisson Regressions

- The *offset*: The Poisson model can be seen not as a model of count data directly but of a *rate*.

$$y_i \sim \text{Poisson}(u_i\theta_i)$$

  where u_i, typically called the exposure (a baseline) and θ_i is the rate at which events (e.g visits) happen. The $\log(u_i)$ is called the offset and added to the regression as `~1` and typically has a fixed coefficient of 1. The offset often represents an attribute of the population, e.g. the size of the population the count data come from. Then the parameters can be interpreted as rates of occurrence in the population.
- *Overdispersion*: Because the Poisson distribution has one parameter, it doesn't have an independently estimable variance σ. This often means that the data are *overdispered* i.e. more variable than the model can handle. This can be adjusted for by using `family="quasipoisson"` or the negative binomial family.
- *Comparison with logistic-binomial models*: When the data are count data but *out of a fixed total* n then logistic models can be used.

Workshop 10: Logistic regression, diagnostics and the APC

Learning outcomes

Adding multiple predictors to a logistic regression in `R`

Predicting probabilities

Coding Average Predictive Comparisons

Packages

```
library(arm)
library(car)
```

Multiple predictors

Today we'll be using the data set "dmd.csv" and run a regression with multiple predictors. There are 209 observations corresponding to blood samples collected in a project to develop a screening program for female relatives of boys with Duchenne Muscular Dystrophy (DMD) a congenital disease. There are 4 questions we need to answer:

1. The program's main goal was to inform a woman of her chances of being a carrier based on serum markers (chemicals in the blood).
2. Also of interest was whether age should be taken into account.

Enzyme levels were measured in known carriers (75 samples) and in a group of non-carriers (134 samples). The first two serum markers, creatine kinase and hemopexin (`ck`,`h`), are inexpensive to measure, while the last, pyruvate kinase (`pk`) is more expensive to measure.

3. We also want to know whether `ck` and `h` are sufficient to predict carrier status or whether `pk` is necessary. In routine blood tests, it is not possible to measure all three, if we measure `ck` and `h` we cannot measure `pk`. However all three were measured for this study.

The water supply in the lab in which the blood samples were being analysed changed during the study. `afterchg=1` indicates which samples were taken after the change and `afterchg=0` which were taken prior to the change.

4. Finally we want to know whether the change in water had an effect on the outcomes

The data look like this:

```
dmd.dat<-read.csv("dmd2.csv",header = T)
head(dmd.dat)
```

```
##   age afterchg ck     h       pk carrier
## 1  27        1 15  87.0 13.50000       0
## 2  39        1 18  95.0 11.29883       0
## 3  58        1 19 100.5 10.89844       1
## 4  22        0 20  72.0 11.89844       0
## 5  32        1 20  77.0 11.00000       0
## 6  22        0 21  74.5 12.19922       0
```

Before we do any regressions let's centre the continuous predictors. This makes sense because the measurements are not taken on new-borns and 0 values are not possible for the serum markers as they exist in some concentration in every person.

```
#write function
cent.pred<-function(v){
  cent.v<-v-mean(v)
  cent.v
}
#center
cent.ck<-cent.pred(dmd.dat$ck)
cent.h<-cent.pred(dmd.dat$h)
cent.age<-cent.pred(dmd.dat$age)
cent.pk<-cent.pred(dmd.dat$pk)
```

Let's start by using two regressions, one with `cent.pk`,

```
dmd.pk.glm<-glm(carrier~cent.age+afterchg+cent.pk,data=dmd.dat,
                family=binomial(link="logit"))
display(dmd.pk.glm)
```

```
## glm(formula = carrier ~ cent.age + afterchg + cent.pk, family = binomial(link = "logit"),
##     data = dmd.dat)
##             coef.est coef.se
## (Intercept) -1.23     0.59
## cent.age     0.18     0.03
## afterchg     0.44     0.63
## cent.pk      0.17     0.04
## ---
##   n = 206, k = 4
##   residual deviance = 150.0, null deviance = 259.9 (difference = 109.9)
```

and another with `cent.ck`, `cent.h`.

```
dmd.ckh.glm<-glm(carrier~cent.age+afterchg+cent.ck+cent.h,data=dmd.dat,
                 family=binomial(link="logit"))
display(dmd.ckh.glm)
```

```
## glm(formula = carrier ~ cent.age + afterchg + cent.ck + cent.h,
##     family = binomial(link = "logit"), data = dmd.dat)
##             coef.est coef.se
## (Intercept) -0.79     0.65
## cent.age     0.14     0.03
## afterchg     0.28     0.64
## cent.ck      0.02     0.01
## cent.h       0.06     0.02
## ---
##   n = 206, k = 5
##   residual deviance = 139.4, null deviance = 259.9 (difference = 120.4)
```

`pk`, `ck` and `h` are all significant, indicating that the models *do* help predict carrier status. We can confirm this using the diagnostics: For both models the difference between the null and residual deviances are above the critical value for the corresponding χ^2 distribution. Specifically:

1. For the regression with `pk` the difference between the residual and null deviance is 109.9. This needs to be compared to the χ^2 distribution on 3 degrees of freedom.

```
qchisq(0.95,3)
```

```
## [1] 7.814728
```

109.9>7.81 so the model with `pk`, `age` and `afterchg` is better than the null model. A similar argument leads us to compare 120.4 (difference between null and residual deviance in model with `ck` and `h`) with:

```
qchisq(0.95,4)
```

```
## [1] 9.487729
```

So the model with `ck`, `h`, `age` and `afterchg` is better than the null model.

However in both cases `afterchg` appears to have little effect in terms of significance Furthermore, `age` is a significant predictor of being a carrier. This answers questions 1,2 and 4 above.

Let's turn to question 3: "Are `ck` and `h` sufficient to predict carrier status, or is `pk` better?" In order to do this, let's compare the classification tables for both models. First let's write a function that produces classification tables.

```
# function arguments create the data frame
ct.op <- function(predicted, observed) {
    df.op <- data.frame(predicted = predicted, observed = observed)
    # create a table
    op.tab <- table(df.op)
    # use the prop.table function to obtain the proportions we need: those who
    # were correctly predicted as 0 @position 1,1 in the table of proportions
    obs0.tab <- round(prop.table(op.tab, 2)[1, 1], 2)
    # those who were correctly predicted as 1 @position 2,2 in the table of
    # proportions
    obs1.tab <- round(prop.table(op.tab, 2)[2, 2], 2)
    # and put them under the table
    op.tab <- rbind(op.tab, c(obs0.tab, obs1.tab))
    # name the rows
    rownames(op.tab) <- c("pred=0", "pred=1", "%corr")
    # name the columns
    colnames(op.tab) <- c("obs=0", "obs=1")
    # return the table
    op.tab
}
```

How well does the model with `pk` predict?

```
#we use as.numeric because the part inside the brackets returns TRUE/FALSE
pred.pk<-as.numeric(dmd.pk.glm$fitted.values>0.5)
#pass the fitted values and the observed values to ct.op
ct.op(pred.pk,dmd.dat$carrier)
```

```
##         obs=0 obs=1
## pred=0 133.00  20.0
## pred=1   6.00  47.0
## %corr    0.96   0.7
```

How about the model with `ck` and `h`?

```
pred.ckh<-as.numeric(dmd.ckh.glm$fitted.values>0.5)
ct.op(pred.ckh,dmd.dat$carrier)
```

```
##         obs=0 obs=1
## pred=0 132.00 19.00
## pred=1   7.00 48.00
## %corr    0.95  0.72
```

Q: Based on the output of `display()` and the tables of predicted vs observed would you recommend the test using `ck` and `h` or the test using `pk`? Explain your answer.

Predicting probabilities and risk ratios

As mentioned before, the logistic function is non-linear which means that a change in one in the value of a predictor is not the same across the range of the predictor. Specifically, `age` going from 20-30 may not result in the same change in the probability of being a carrier as `age` going from 30-40.

Using plots, explain the statement above.

One way to see this numerically is to look at *risk ratios* or *relative risks*. We've done this by hand for a single predictor, but how can we get R to do it for us in the context of multiple predictors?

Consider the model with ck, h and age:

```
dmd.glm.apc<-glm(carrier~age+ck+h,data=dmd.dat,family=binomial)
display(dmd.glm.apc)
```

```
## glm(formula = carrier ~ age + ck + h, family = binomial, data = dmd.dat)
##             coef.est coef.se
## (Intercept) -11.62     1.94
## age           0.14     0.03
## ck            0.02     0.01
## h             0.06     0.02
## ---
##   n = 206, k = 4
##   residual deviance = 139.6, null deviance = 259.9 (difference = 120.2)
```

What is the change in probabilities associated with a change in *age* of

a) 20 to 25 and
b) 40 to 45

for a woman with mean ck (90.19) and mean h (86.37)? Ages over 45 are not of interest because the probability of women above 45 having children is very low.

1. We create the data frame we want to use for prediction: Note that you could create this manually in excel and save it as a csv file and load it into R if you prefer.

```
new.dmd.dat<-data.frame(age=c(20,25,40,45),ck=mean(dmd.dat$ck),h=mean(dmd.dat$h))
new.dmd.dat
```

```
##   age       ck       h
## 1  20 90.18689 86.36711
## 2  25 90.18689 86.36711
## 3  40 90.18689 86.36711
## 4  45 90.18689 86.36711
```

2. Next we use the predict function, this gives us the logits associated with the values above.

```
pred.logit<-predict(dmd.glm.apc,new.dmd.dat)
#pred.logit
```

3. To obtain probabilities, we can either use the formula or we can ask R to do it: (remember the invlogit function is in the arm package.)

```
pred.probs<-invlogit(pred.logit)
#pred.probs
```

To see these together a small data frame:

```
logreg<-data.frame(logits=pred.logit,probs=pred.probs)
cbind(new.dmd.dat,logreg)
```

```
##   age       ck       h     logits     probs
## 1  20 90.18689 86.36711 -2.1832492 0.1012648
## 2  25 90.18689 86.36711 -1.4992675 0.1825348
## 3  40 90.18689 86.36711  0.5526776 0.6347566
## 4  45 90.18689 86.36711  1.2366593 0.7749820
```

We see that the probabilities of being a carrier are very low for younger women and increase substantially for older women with mean values of ck and h.

4. Now calculate the risk ratios:

```r
print("risk ratio age:20-25, mean ck and h")
```

```
## [1] "risk ratio age:20-25, mean ck and h"
```

```r
as.numeric(pred.probs[2]/pred.probs[1])
```

```
## [1] 1.802549
```

```r
print("risk ratio age:40-45, mean ck and h")
```

```
## [1] "risk ratio age:40-45, mean ck and h"
```

```r
as.numeric(pred.probs[4]/pred.probs[3])
```

```
## [1] 1.220912
```

```r
#as.numeric means we don't get the data frame formatting
```

We see that the probability of being a carrier increases by 80% when going from the age of 20 to 25 whereas is only increases by 22% when going from 40-45 years of age for women with average `ck` and average `h` levels. These figures look alarming without the probabilities themselves to relate to so display risk ratios alongside probabilities.

Q: Why do is it a good idea to use the mean (or median) value for the predictors we are not interested in comparing?

Let's see how those values are affected if we look at women with high `ck`: 73 is the 75th quantile.

```r
new.dmd.dat<-data.frame(age=c(20,25,40,45),ck=73,h=mean(dmd.dat$h))
pred.logit<-predict(dmd.glm.apc,new.dmd.dat)
pred.probs<-invlogit(pred.logit)
logreg<-data.frame(logits=pred.logit,probs=pred.probs)
cbind(new.dmd.dat,logreg)
```

```
##   age ck        h     logits      probs
## 1  20 73 86.36711 -2.5468421 0.07263892
## 2  25 73 86.36711 -1.8628604 0.13437000
## 3  40 73 86.36711  0.1890847 0.54713083
## 4  45 73 86.36711  0.8730663 0.70538334
```

```r
print("risk ratio age:20-25, ck=73 and  mean h")
```

```
## [1] "risk ratio age:20-25, ck=73 and  mean h"
```

```r
as.numeric(pred.probs[2]/pred.probs[1])
```

```
## [1] 1.849835
```

```r
print("risk ratio age:40-25, ck=73 and  mean h")
```

```
## [1] "risk ratio age:40-25, ck=73 and  mean h"
```

```
as.numeric(pred.probs[4]/pred.probs[3])
```

```
## [1] 1.289241
```

For women with high`ck` the situation appears to be only slightly worse in terms of risk ratios (85 and 29% for younger and older women respectively) but is actually slightly better in terms of the probabilities (13 and 71% probability of being a carrier at 25 and 45 years respectively).

Average predictive comparisons.

The APC is a generalisation of the approach above where we input predictor means for the predictors that are not the focus of our interest. One big difference is that we look at *differences* between probabilities and *not ratios*.

Let's apply this technique to the DMD data set. We should not use centered predictors for this. Also, remove `afterchg` as it is not significant.

For age:

```
dmd.glm.apc<-glm(carrier~age+ck+h,data=dmd.dat,family=binomial)

b<-coef(dmd.glm.apc)
hi<-45 #actual highest is 61 but women mostly have children before 45
lo<-20
delta.age<-with(dmd.dat,(invlogit(b[1]+b[2]*hi+b[3]*ck+b[4]*h)
                         -invlogit(b[1]+b[2]*lo+b[3]*ck+b[4]*h)))
print(mean(delta.age))
```

```
## [1] 0.4563661
```

This means that the average predictive difference in probability of being a carrier associated with an increase over the observed range of years is 46% other things remaining equal. Remember that the range of age is from 20-61 but we only consider 20-45 as child bearing ages.

For `ck`:

```
hi<-1000
lo<-15
delta.ck<-with(dmd.dat,(invlogit(b[1]+b[2]*age+b[3]*hi+b[4]*h)
                        -invlogit(b[1]+b[2]*age+b[3]*lo+b[4]*h)))
mean(delta.ck)
```

```
## [1] 0.826888
```

For `h`:

```
hi<-118
lo<-34
delta.h<-with(dmd.dat,(invlogit(b[1]+b[2]*age+b[3]*ck+b[4]*hi)
                       -invlogit(b[1]+b[2]*age+b[3]*ck+b[4]*lo)))
print(mean(delta.h))
```

```
## [1] 0.4494374
```

Q: Interpret the APC for `ck` and `h`.

More materials for the course for reference or to do in your own time

1. Model building
2. Guide to model assessment
3. ggplot extra
4. Least squares estimation Exercise
5. Loops, functions and apply Exercise
6. Categorical variable interactions Exercise
7. Omitted variables Exercise

Model Building

Now that you have all the main tools used in regression we can consider the "bigger picture" which is how to build regression models in statistcs.

First off: **statistical modelling is something of an art**. If you give the same (complex) data set to two statisticians you may find that they come up with two different models. If they know what they are doing they won't be very different but they will be differences – so there is not necessarily one right answer. You *must* learn to use your judgement (and interactions with team members when working together) to make decisions about what steps to take in an analysis. So you *must* be able to defend any steps you take. This means you *cannot* analyse the data without thinking carefully about which steps to take. *If you are not thinking about what the results mean at every step and are focussing only on diagnostics or plots then you are not doing it correctly!*

Typically when faced with a new data set and a specific research question a statistician will do the following before starting to run regressions:

- Think about what relationships you expect there to be between the variables (i.e. you expect age to be positively correlated with salary)
  - but be prepared to reconsider your expectations when more variables come into play i.e. there may be confounding/non-linearity
- Think carefully about whether there may be omitted predictors i.e. predictors you would like to have observed but haven't.
  - Is it possible to obtain appropriate data on these from another source, do the experts have these data?
  - If it is not possible to obtain data on these confounders then how may omitting them from the analysis bias the results
- If you are working with an expert then make sure you know what they think should be going on
  - beware though of assuming that the expert will be completely correct as they may have pre-conceptions
- Look at the summary of the data
- Check that you know which variables are what (type, role)
- Plot those that are most relevant (typically the outcome vs the predictors)
- Check these plots make sense (are they non-linear, do the signs make sense?)
- Are there potential outliers? No need to do anything yet but bear them in mind.

I cannot emphasise how important it is to have thought hard about the problem and the available data *BEFORE* you start running regressions.

Then start the analyses and do approximately this (not necessarily but often in this order) :

1. Start off with a regression with all the variables that for non-statistical reasons (i.e. the expert says so) make sense as predictors
2. Sometimes it makes sense to combine predictors in a score (as was done with the Study.Skills)
3. Transform variables where this makes sense (e.g. if you want to be able to interpret the intercept or if there is clear evidence of non-linearity)
4. If after an initial regression some predictors have large effects (check the standardised coefficients) check if interactions are also important
5. Check overall model diagnostics and run the residual plots
6. Identify outliers (as detailed in WorkShop 5)
7. Run some cross-validation

Note: There is typically no need to perform steps 6 and 7 for every single model.

At various points you will have to decide whether to include a predictor or not. Do the following:

- If a predictor is non-significant at the 5% level but the sign is correct leave it (with some caveats see below)

- If a predictor is non-significant at the 5% level and the sign is not correct remove it
- If a predictor is significant at the 5% level and does not have the expected sign then *think hard*! Why is this happening? Are there lurking/confounding variables? Have you made a mistake? If you think there are Consider that your expectations may have been wrong and that you need to re-think.
- If a predictor is significant at the 5% level and has the expected sign then keep them

Caveats: If there are a lot of non-significant predictors then think if you need another one. If the predictor is a level of a categorical variable consider combining some levels of the predictor.

During the course of an analysis you will typically try different things. Some of these are:

- Add and remove predictors
- Combine many-levelled categorical variables
- Try different transforms
- Turn continuous variables into categorical variables
- Analyse subsets of the data separately (e.g. men/women, interested/uninterested)
- Remove outliers
- Run cross-validation

The aim is to get a model or models that are best at predicting and/or best at explaining the relationships between the variables given the data you have and the research question. Remember that the best models you can obtain using regression methods may not be very good.

Some rules of thumb

1. A model with parameters that are easier to interpret is often preferable to one that has hard to interpret parameters
2. A model which leads to better predictions is often preferable to one with worse predictions
3. A simple model is often preferable to a more complex model
4. A model with better diagnostics is often preferable to one with worse ones

The 4 criteria above are often at odds with one another especially when the improvement is not large (e.g. cross-validation predictions aren't that much better). For the course-work I am happy for you to present 2 models with a preference for one of them.

This is not always possible in a working context so decisions need to be made and justified – however you should say that the model is not very good if that is the case. For instance you can discuss the results of cross-validation in lay terms saying that the predictions are not very good. An employer will want to know how certain you are of your model.

Guide to model assessment

This is a brief guide to what things to do/consider when assessing model fit. Let's use the Study Habits data without the outliers.

```
Study_Habits<-read.csv("Stats_study_habits_noout.csv",header=T)
head(Study_Habits,2)
summary(Study_Habits)
```

Note that many of the binary/categorical predictors are coded as numbers. This means that you need to `as.factor` them. Cohort is either 2012 or 2013 and is an indicator of whether the student that responded to the questionnaire on study skills took the course in 2012 or 2013. Gender is "M" and "F". A.level is the grade in Maths/Stats on a scale from 0 to 6 and is considered to be a continuous variable.

```
grade.all.lm<-lm(Grade~as.factor(Cohort)+as.factor(Interesting)+as.factor(Enjoy)
                 +as.factor(Social.Net)+Gender+A.Level+Study.Skills,data=Study_Habits)
display(grade.all.lm)
```

```
## lm(formula = Grade ~ as.factor(Cohort) + as.factor(Interesting) +
##     as.factor(Enjoy) + as.factor(Social.Net) + Gender + A.Level +
##     Study.Skills, data = Study_Habits)
##                          coef.est coef.se
## (Intercept)              20.61    20.08
## as.factor(Cohort)2013     4.33     3.05
## as.factor(Interesting)1  10.57     3.59
## as.factor(Enjoy)1         2.38     3.58
## as.factor(Social.Net)2    2.48     5.16
## as.factor(Social.Net)3   -0.05     4.89
## GenderM                  -4.03     3.20
## A.Level                   0.95     3.24
## Study.Skills              1.29     0.33
## ---
## n = 85, k = 9
## residual sd = 13.86, R-Squared = 0.39
```

What to do with the output of `display()`

Q1: For each predictor in the model:

1. Interpret the coefficients associated with it
2. Check the signs of the coefficients make sense
3. Determine whether the predictor is significant at the 5% level

Q2: Is the model a good fit? Consider the following:

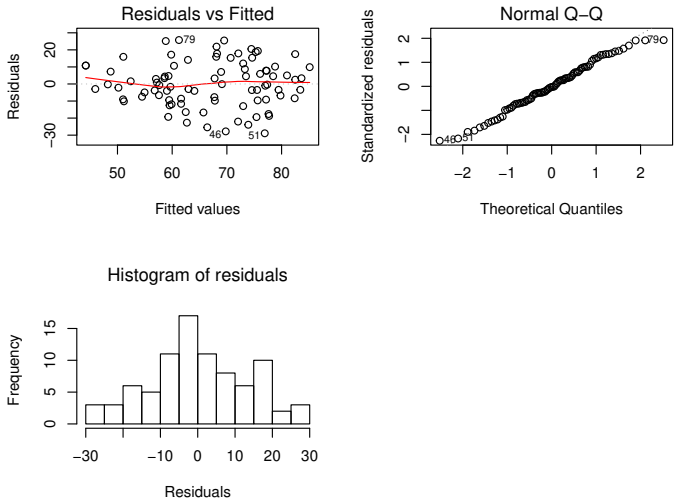
1. R^2
2. The residual sd (check the absolute range of `Grade` using `abs(range(Study_Habits$Grade))`)
3. The significance of the coefficients. If there are too many non-significant coefficients it makes sense to get rid of some especially if the sign does not makes sense. Which 2 would you get rid of and why?

(If you want to see the Adjusted R^2 or the F-statistic, look at `summary()`)

It's not all about the diagnostics! You need to look at residual plots too!

The same residual plots are relevant for multiple linear regression as for simple linear regression.

```r
par(mfrow=c(2,2))
plot(grade.all.lm,which=c(1,2))
hist(grade.all.lm$residuals,main="Histogram of residuals",font.main=1,xlab="Residuals")
```



Q3: Based on the plots can we conclude that the residual assumptions of constant variance and normality of residuals hold? Look for the following:

1. Does the fitted vs residual plot look like random scatter (no trends, curvature, funnel shape)?
2. Do the points in the Q-Q plot sit approximately on the diagonal (no bow/s shapes)
3. Does the histogram of the standardised residuals look approximately normal (no bimodality, skewness, kurtosis)?

Q4: If some of the predictors are non-significant you can consider removing them – I'm not a big fan of doing this especially if the non-significance is borderline (e.g. a p-value of 0.06) and when the sign of the predictor has the right sign/size. If you do remove them, does anything change?

Q5: Should you add interactions? What happens if you do?

ggplot Examples

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

Learning outcomes

In this section there are example of plots using the `ggplot2` package.

1. Basic box and scatterplots.
2. Arranging plots using `gridExtra`
3. Adding points to a scatterplot
4. Using colour, shape and fill to add information about categorical variables to a box and scatterplot
5. Adding a straight line to a scatterplot
6. Adding a linear model to a scatterplot in simple linear regression
7. Adding multiple linear models to a scatterplot in multiple linear regression
    a. Without an interaction
    b. With an interaction
8. Using `facet_wrap` to display data across categories
9. barplots for cross tabulation
10. Special plots

Throughout we'll use the same data: The Boston Housing data. This has been modified so that `ZN`, one of the predictors is a 3 level categorical predictor.

```
Boston.dat<-read.csv("Bostonggplot.csv", header=TRUE)
head(Boston.dat,3)
```

```
##      CRIM     ZN INDUS CHAS   NOX    RM  AGE    DIS RAD TAX   PT LSTAT
## 1 0.00632 middle  2.31    0 0.538 6.575 65.2 4.0900   1 296 15.3  4.98
## 2 0.02731   Zero  7.07    0 0.469 6.421 78.9 4.9671   2 242 17.8  9.14
## 3 0.02729   Zero  7.07    0 0.469 7.185 61.1 4.9671   2 242 17.8  4.03
##     MV
## 1 24.0
## 2 21.6
## 3 34.7
```

1. Basic scatterplots

Let's consider the continuous predictor `LSTAT`.

```
#aes stands for "aesthetics"
#without +geom_point() the plot is empty
ggplot(data=Boston.dat, aes(x=LSTAT, y=MV))+geom_point()
```

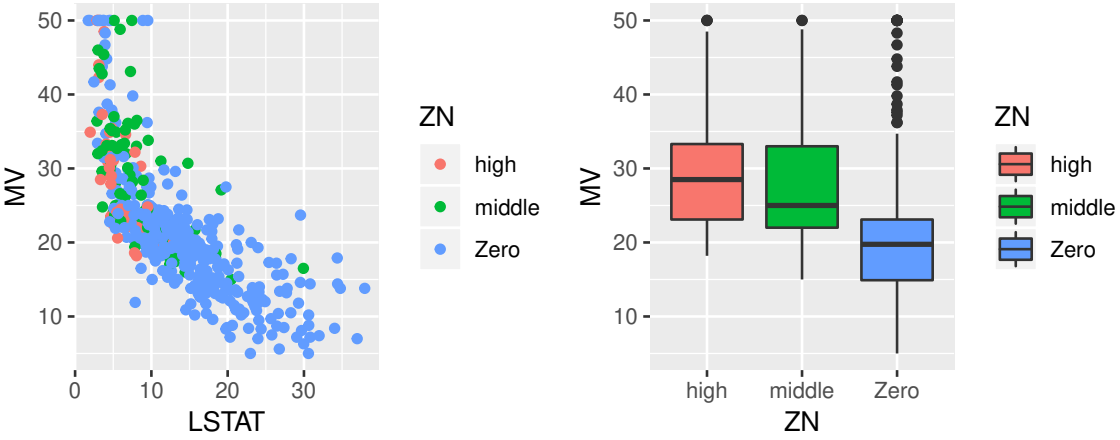
```
#aes stands for "aesthetics"
#without +geom_boxplot() the plot is empty
ggplot(data=Boston.dat, aes(x=ZN, y=MV))+geom_boxplot()
```

To avoid using a lot of space, use `grid.arrange()` in the `gridExtra` package.

```
#scatterplot
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV))+geom_point()
#boxplot
p2<-ggplot(data=Boston.dat, aes(x=ZN, y=MV))+geom_boxplot()
#place them side by side (nrow=1) or one above the other (nrow=2)
grid.arrange(p1,p2,nrow=1)
```

Most changes you want to make to the scatter/boxplot are now just a matter of adding something to `p1`/`p2` . Some changes have to be in the main body of `ggplot()` in `aes()`.

```
#scatterplot with different colours for different levels of ZN
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV, colour=ZN))+geom_point()
#boxplot with colour in the boxplots
p2<-ggplot(data=Boston.dat, aes(x=ZN, y=MV,fill=ZN))+geom_boxplot()
#place them side by side (nrow=1) or one above the other (nrow=2)
grid.arrange(p1,p2,nrow=1)
```

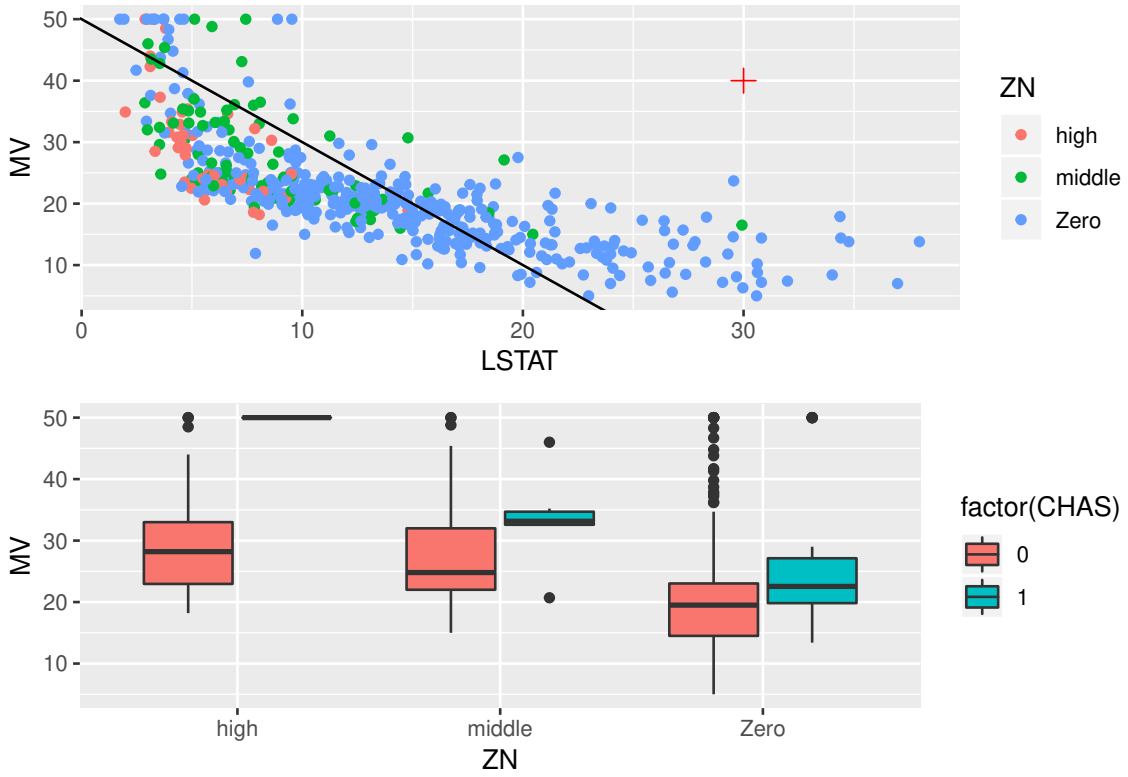


Let's add

1. a point and a straight line to the scatterplot
2. An interaction to a boxplot

```
#basic scatterplot
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV, colour=ZN))+geom_point()
#creating point
new.point<-data.frame(LSTAT=30,MV=40,ZN="Zero")
#adding point in red with a larger size and a different shape
p1<-p1+geom_point(data=new.point, color="red", size=3, shape=3)
#adding a straight line with intercept 50 and slope -2
p1<-p1+geom_abline(intercept=50, slope=-2)

#basic boxplot with ZN grouped by CHAS
```

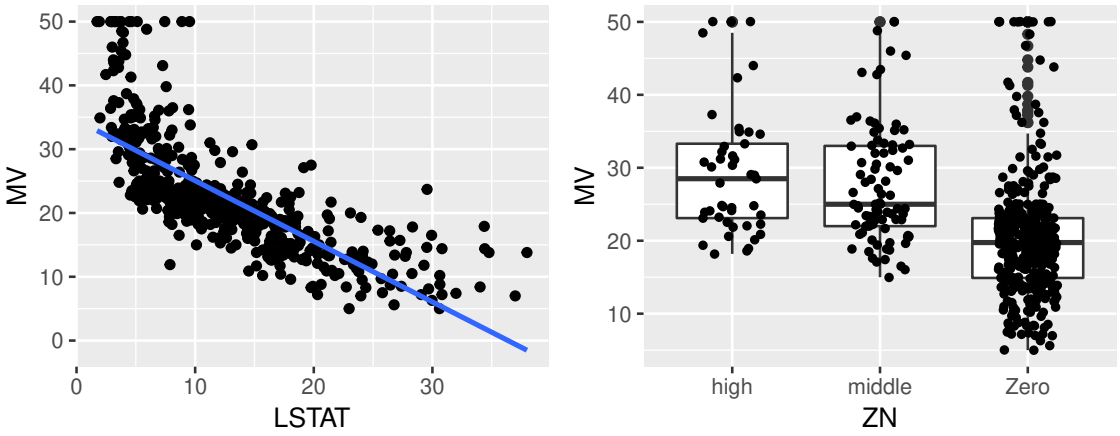
```
p2<-ggplot(data=Boston.dat, aes(x=ZN, y=MV,fill=factor(CHAS)))+geom_boxplot()
#over two rows for grid.arrange
grid.arrange(p1, p2, nrow=2)
```



Next, let's add

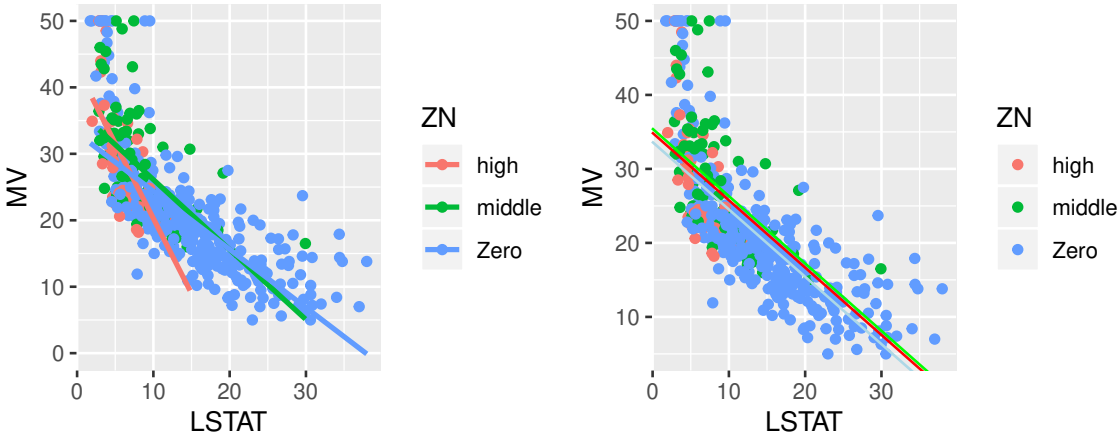
1. A line of best fit to a scatterplot
2. Add points to the boxplot

```
#basic scatterplot
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV))+geom_point()
#adding a line of best fit using a linear model (se=FALSE means no error bars)
p1<-p1+geom_smooth(method="lm",se=FALSE)

#basic boxplot with ZN
p2<-ggplot(data=Boston.dat, aes(x=ZN, y=MV))+geom_boxplot()
#add the points and jitter them so they group together too much
p2<-p2+geom_jitter(shape=16, position=position_jitter(0.2))

grid.arrange(p1, p2, nrow=1)
```



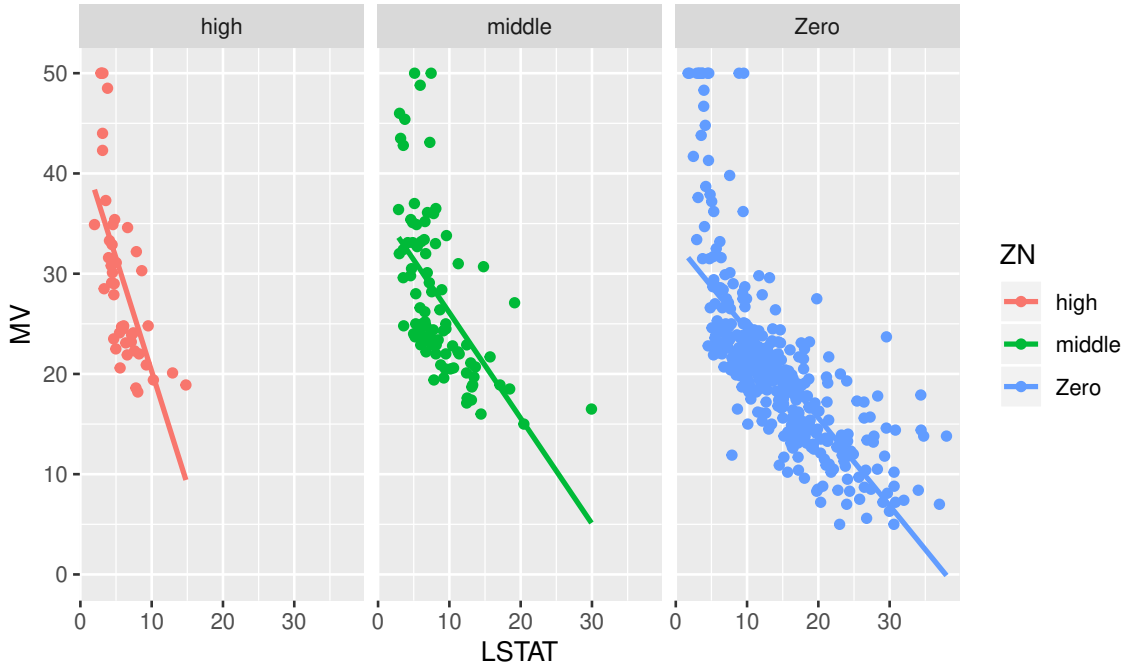
If we add a `geom_smooth` to the plot with `colour=ZN`, three non-parallell lines appear. Basically `geom_smooth` uses the interaction model to plot the lines. If we want to show what the lines look like without an interaction then we need to use `geom_abline()`

```
#basic scatterplot
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV, color=ZN))+geom_point()
#adding a line of best fit using a linear model (se=FALSE means no error bars)
p1<-p1+geom_smooth(method="lm",se=FALSE)

#run the regression:
lm.plot<-lm(MV~LSTAT+ZN, data=Boston.dat)
#basic  scatterplot
p2<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV, colour=ZN))+geom_point()
#add three lines
p2<- p2+ geom_abline(intercept = coef(lm.plot)[1], slope = coef(lm.plot)[2], color="red")
p2<- p2 + geom_abline(intercept = (coef(lm.plot)[1]+ coef(lm.plot)[3]), slope = coef(lm.plot)
p2<- p2 + geom_abline(intercept = (coef(lm.plot)[1]+ coef(lm.plot)[4]), slope = coef(lm.plot)

grid.arrange(p1, p2, nrow=1)
```



Another way of displaying data by levels of a categorical variable is to use `facet_wrap()`.

```
#basic scatterplot
p1<-ggplot(data=Boston.dat, aes(x=LSTAT, y=MV, color=ZN))+geom_point()
#adding a line of best fit using a linear model (se=FALSE means no error bars)
p1<-p1+geom_smooth(method="lm",se=FALSE)
p1<-p1+facet_wrap(~ZN)
p1
```


high
middle
Zero
MV
LSTAT
0
10
20
30
40
50
ZN
high
middle
Zero

Least Squares optimisation Excercise

Instructions

We've seen how to derive the least squares parameter estimates using calculus. It is also possible to do it numerically by using an optimisation routine. Optimisation is typically used to minimise or maximise a formula that is based on data

This homework contains the script for and teaches you how to use `function` and write your own function to optimise the SSE in `R`

The function

First we write the function we need to optimize.

```
squaredres<-function(x,w,h){
res<-w-(x[1]+x[2]*h)
sumsqres<-sum(res^2)
return<-sumsqres
}
```

- First line declares the function, its name is `squaredres`
- Each function has its own *internal environment* : variables and potentially functions *inside* it
- Inside the `function()` brackets there are *arguments* : the variables that the function gets passed from outside. In our case
  - `x` is the vector of parameters to minimise (β_0 and β_1)
  - `h` vector of the values of the predictor variable (the x-axis) and
  - `w` are the values of the outcome variable (the y-axis)
- The next line is the formula for the residuals `r<-w-(x[1]+x[2]*h)`
- Then comes the sum of the squares `sumsqres<-sum(r^2)`
- Finally `return` tells the function what to return to us. In this case it is `sumsqres`
- names in the function over-ride names in the global environment: this means that if I re-define `x` inside the function then it takes that to be the true value and ignores previous definitions of `x`.

Using the function

On Moodle are 5 datasets called "lsehwk_.csv" where _ represents a number from 1-16. Choose one at random (but remember which one) and load it into `R`. The 5 data sets are subsets of different sizes (10, 15, 20, 25, 30) of the full dataset which has 38 points. Note that the script below refers to the full dataset that you do not have access to.

Now let's use the function `squaredres`. We pass it as an argument to the `R` function `optim()` which as default finds the parameters that minimise the value of the function.

- `optim` assumes that `x` in the function are the parameters that can be changed to optimise the function. This is a bit confusing because we have 2 `x`s. However the name in the function over-rides the name outside the function.
- It takes the starting values as `par(c(0,5))` in this case our guesses for the intercept and the slope
- `fn` is the function it needs to optimise.
- `h` and `w` are additional arguments for the function.

I am only interested in the parameter values so I say `$par` at the end of the function call.

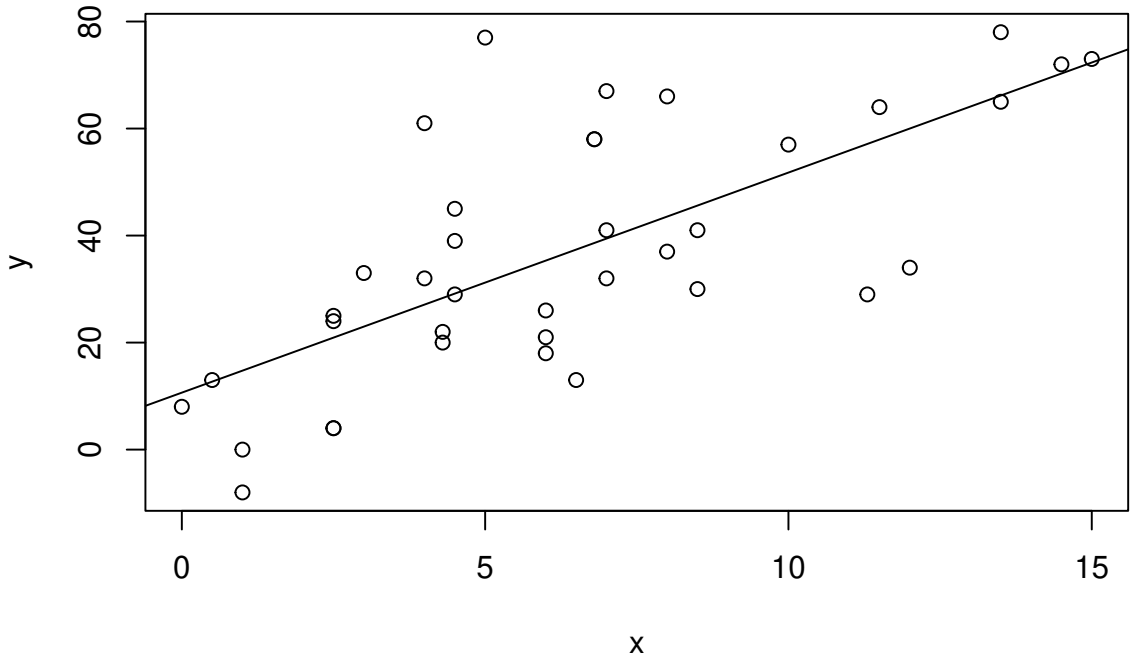
```
hats<-optim(par=c(0,5),fn=squaredres,w=lsehwk.dat$y, h=lsehwk.dat$x)$par
hats
```

```
## [1] 10.644953  4.112922
```

`$par` are the values of $\hat{\beta}_0$ and $\hat{\beta}_1$. 10.6 is the *intercept* and 4.11 is the *slope* of the line.

Plot the line using `abline`

```
with(lsehwk.dat,plot(x,y))
abline(a=hats[1],b=hats[2])
```



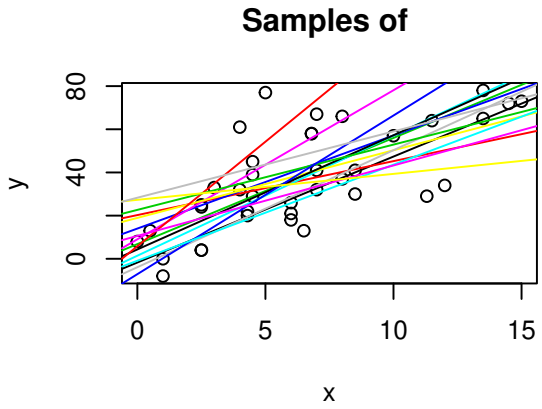
You should find that your parameters are not the same as mine. Why do you think that is? Think about your sample and its size. Compare your plot to mine.

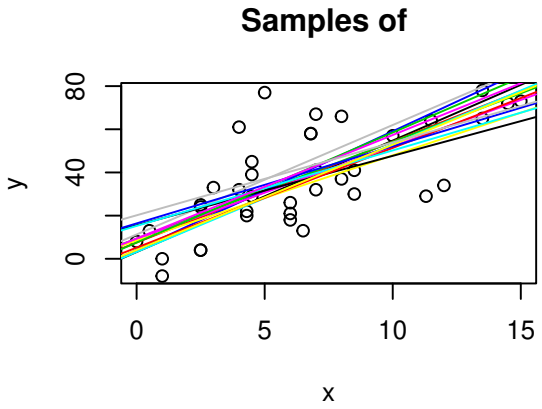
If you have time, repeat the process with all 5 data sets.

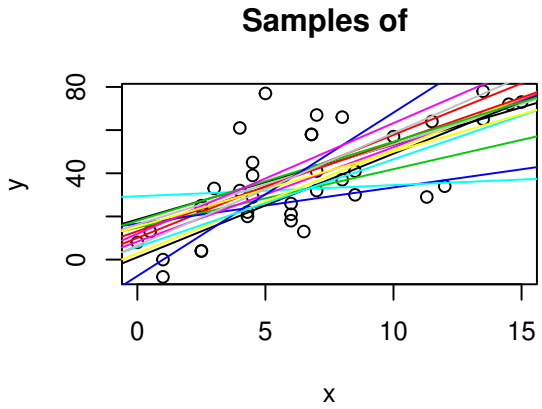
Sample variation

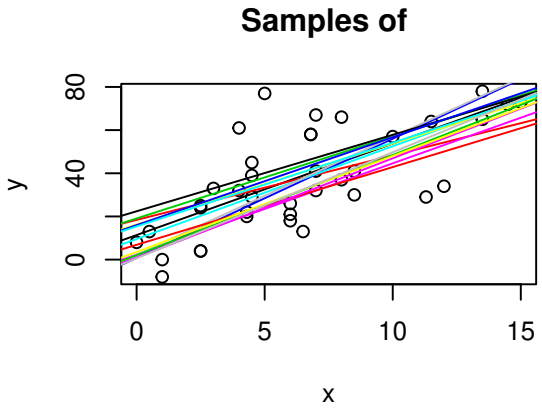
Typically every sub-sample results in a different line of best fit.

- Would you expect to see more variability in the slopes and intercept for a set of subs-samples of 10?
- How about 30 or 5?
- Fill in 5,10,15 and 30 in the plots below. What does this tell you about how reliable analyses can be?


Samples of
y
0
40
80
0
5
10
15
x


Samples of
y
0
40
80
0
5
10
15
x


Samples of
y
0
40
80
0
5
10
15
x


Samples of
y
0
40
80
0
5
10
15
x

Introduction to Functions and Loops

Functions

Functions are pre-packaged bits of code that can be used again and again without having to re-write the code. `R` has many many in-built functions and many more in external libraries that can be installed and loaded. There are functions for almost every common statistical technique.

We have already encountered a number of `R` functions and you will be writing one later in the term as part of a formative assessment. Functions work as follows:

1. Functions have a *name*: e.g. `plot`
2. Functions have *arguments*: e.g. `x` (x-axis variable), `y` (y-axis variable)

i. Some arguments are required and the function call will return an error without them: E.g. `x` in `plot`
ii. Some arguments are optional and the function call will run without them: E.g. `col` (colour of points) in `plot`

iii. The optional arguments tend to have defaults, i.e. values that the function assumes if no other instruction is given: E.g. the default for `col` is black. 3.Functions have *output*: E.g. a plot of `x` vs `y`.

For the function `mean` what are the required and optional arguments? What is the output? (Hint: just type "mean(" into the console and then tab and see what happens)

Writing our own `mean` function

The formula for the mean of a vector is:

Our function: `mean.2` will take a vector `vec` as argument and will output the mean

```
mean.2<-function(vec){
  r<-sum(vec)/length(vec)
  r
}

new.vec<-c(1,2,3,4,5)
mean.2(new.vec)
```

```
## [1] 3
```

```
new.vec<-c(1,2,3,4,NA)
mean.2(new.vec)
```

```
## [1] NA
```

```
#new.vec<-c(1,2,3,4,"Hi")
#mean.2(new.vec)
```

Notice that we are cheating a bit in that we are calling two `R` functions: `sum` and `length`. Our function takes advantage of the error codes and defaults of these functions.

Loops

Loops are used in coding when we want to do the same thing (e.g. get the mean) for each element (typically a row or column) of data. `R` has two main loop functions `for` and `while`. `for` does the same thing to each element until there are no more elements, `while` does the same thing to each element until a certain criterion has been reached. `for` loops are more common and we'll cover them briefly.

A simple `for` loop: (pat attention to where you put commas in the square brackets)

```
#create an array
small.array<-array(dim=c(6,3))
small.array[,1]<-c(1,2,3,4,5,6) #seq(1:6)
small.array[,2]<-c(2,2,2,2,2,2) #rep(2,6)
small.array[,3]<-c(-1,-2,-3,-4,-5,-6) #seq(1:6)*-1
small.array
for(i in 1:6){
  mn<-mean(small.array[i,])
  print(mn)
}
for(i in 1:3){
  mn<-mean(small.array[,i])
  print(mn)
}
```

built-in functions that do what a loop does

When the elements are many `for` loops can be very time/resource consuming. There are a few `R` functions that can replace some simple `for` loops.

```
rowMeans(small.array)
colMeans(small.array)
```

However you will often want to do something to each variable in your data that is specific to your situation and that hasn't been coded by anyone before.

`apply` and family

The `apply` function (and its relatives, `sapply`, `lapply`, `tapply`) are useful in contexts where you need to apply a function to a dataset with either many variables or many rows. These are not easy functions to get your head around and there are competing tools in `R` (e.g. the `dplyr` and tidyverse libraries) that are more intiutive. However these take time to learn and are for more advanced programmers.

`apply` takes 3 arguments:

- the data set,
- 1 if you want to apply the function to the *rows* and 2 if you want to apply it to the *columns*,
- the name of the function you want to apply

```
apply(small.array, 1, mean)
apply(small.array, 2, mean)
```

Interactions between categorical variables Exercise

Sara Geneletti

Question 1

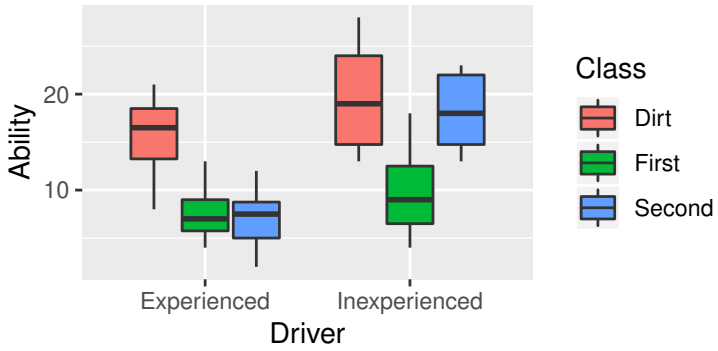
First load the `ggplot2` library and the data. The data is on the Moodle page in the top part and is called drivers2.csv. It is simlar to drivers but has twice the sample size. If you are using your laptop remember you need to install the package first. Go to the packages tab on the bottom right hand corner of RStudio and click install. Type "ggplot2" in the textbox prompt. Enter.

```
library("ggplot2")
drivers.dat<-read.csv("Quizzes/drivers2.csv",header=TRUE)
summary(drivers.dat)
```

```
##            Driver      Class       Ability
##  Experienced  :24   Dirt  :16   Min.   : 2.00
##  Inexperienced:24   First :16   1st Qu.: 7.75
##                     Second:16   Median :13.00
##                                 Mean   :13.00
##                                 3rd Qu.:17.25
##                                 Max.   :28.00
```

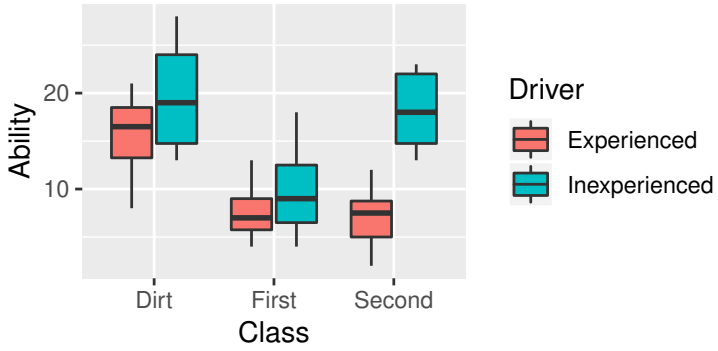
In this plot the x-axis has Driver and for each Class there is a different boxplot.

```
ggplot(aes(y=Ability,x=Driver,fill=Class),data=drivers.dat)+geom_boxplot()
```



Do the same thing except with Class on the x-axis and two boxplots representing the two levels of Driver for each Class.

```
ggplot(aes(y=Ability,x=Class,fill=Driver),data=drivers.dat)+geom_boxplot()
```



Based on these plots would you say Class and Driver interact w.r.t Ability? In particular which levels of Class and Driver?

Answer Class=Second and Driver=Inexperienced. This is because for the first boxplot the relationship between Dirt and First is similar for both levels of Driver but not for Second. Similarly in the second box plot the relationship between experienced and inexperienced are similar in Dirt and In First but different in Second

Run the regression of Ability on both Class and Driver. Are your suspicions confirmed? Usa Anova() to check. ANOVA is useful when there are categorical predictors with multiple categories. Anova() summarises the influence of the whole predictor rather than the significance of each level individually as with display(). It performs a parial F-test.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:arm':
##
##     logit
```

```
drive.lm<-lm(Ability~Class*Driver,data=drivers.dat)
display(drive.lm)
```

```
## lm(formula = Ability ~ Class * Driver, data = drivers.dat)
##                                coef.est coef.se
## (Intercept)                    15.38     1.64
## ClassFirst                     -7.62     2.32
## ClassSecond                    -8.38     2.32
## DriverInexperienced             4.37     2.32
## ClassFirst:DriverInexperienced -2.12     3.28
## ClassSecond:DriverInexperienced  6.75     3.28
## ---
## n = 48, k = 6
## residual sd = 4.63, R-Squared = 0.57
```

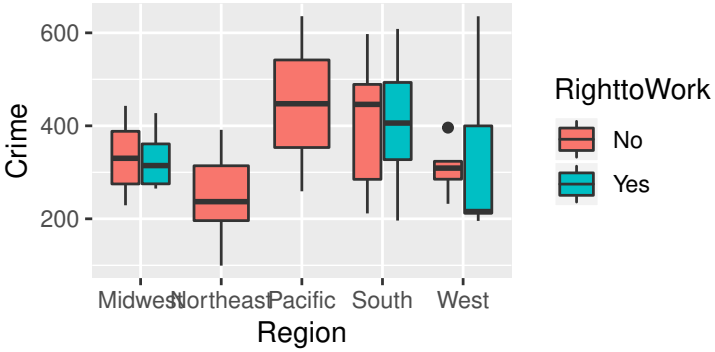
```
Anova(drive.lm)
```

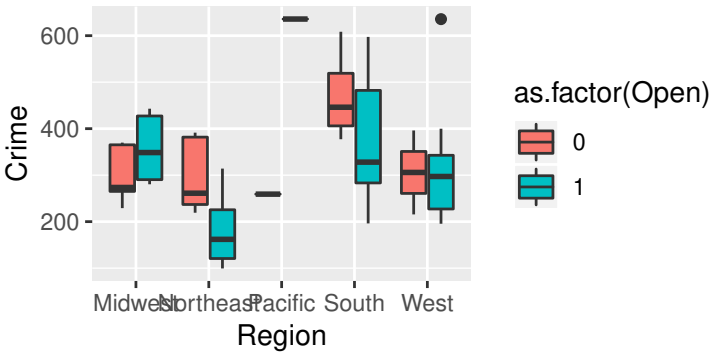
```
## Anova Table (Type II tests)
##
## Response: Ability
##              Sum Sq Df F value    Pr(>F)
## Class        608.37  2 14.1679 1.984e-05 ***
## Driver       420.08  1 19.5658 6.755e-05 ***
## Class:Driver 171.79  2  4.0007   0.02568 *
## Residuals    901.75 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

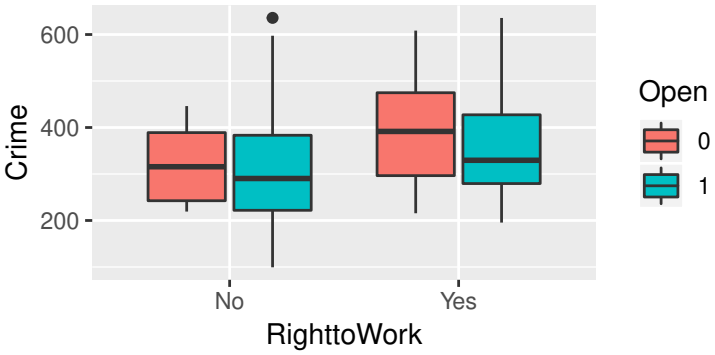
It is often the case that when the sample is too small then even if there looks like there should be an interaction from the boxplot, there isn't because of the size of the sample. *However you can pick it up in the Anova()*

Question 2

For the data gunUSA.csv also on the Moodle page look at whether there is a potential interaction between Region and RightoWork (whether Unions are legal and Open (whether guns are for sale without special permits). The outcome is Crime. For Open you need so specify as.factor(Open) in the ggplot2 function. Below are the plots so that you can see whether you are getting the correct plots.







For Open, can you figure out how to change the label for the Legend so it says Open only? The easiest way is to Google it as the ggplot function is very complex and has many arguments.

Q: For which predictors do you think there may be an interaction? Look also at the tables below. If some cells have very few data then even if there is potentially an interaction in real life, the data aren't sufficient to identify it. See for example Region/RighttoWork. Some regions have no "Yes" at all.

```
with(guns.dat,table(Open, RighttoWork))
```

```
##     RighttoWork
## Open No Yes
##    0 10  10
##    1 15  15
```

```r
with(guns.dat, table(Open,Region))
```

```
##     Region
## Open Midwest Northeast Pacific South West
##    0       5         5       1     7    2
##    1       5         4       1    12    8
```

```r
with(guns.dat, table(Region, RighttoWork))
```

```
##            RighttoWork
## Region      No Yes
##   Midwest    4   6
##   Northeast  9   0
##   Pacific    2   0
##   South      5  14
##   West       5   5
```

Run three regressions of Crime on each set of interactions and run anova() on them. Do your results confirm your inferences from the boxplots?

```
## Note: model has aliased coefficients
##       sums of squares computed by model comparison
```

```
## Anova Table (Type II tests)
##
## Response: Crime
##                     Sum Sq Df F value  Pr(>F)
## Region              153726  4  2.5804 0.05095 .
## RighttoWork            219  1  0.0147 0.90413
## Region:RighttoWork    1144  2  0.0384 0.96236
## Residuals           625534 42
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Anova Table (Type II tests)
##
## Response: Crime
##             Sum Sq Df F value   Pr(>F)
## Region      192216  4  4.0456 0.007571 **
## Open          9669  1  0.8140 0.372346
## Region:Open 142104  4  2.9909 0.029907 *
## Residuals   475124 40
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Anova Table (Type II tests)
##
## Response: Crime
##                   Sum Sq Df F value Pr(>F)
## Open                3681  1  0.2195 0.6416
## RighttoWork        32722  1  1.9513 0.1692
## Open:RighttoWork    5320  1  0.3172 0.5760
## Residuals         771402 46
```

Omitted variables

Omitted or lurking variables

Let's look at weekly pay data (£). Predictors are age (always over 18), house ownership, gender (women=1) and Part-time status. A person is said to work part-time if they work less than 20 hours a week.

```
Weekly_Pay<-read.csv("Confound_Weekly_Pay.csv",header = TRUE)
head(Weekly_Pay)
```

```
##   age  Ptstatus gender WeeklyPay
## 1  20 part-time      F        30
## 2  42 full-time      F       116
## 3  59 part-time      F        60
## 4  63 part-time      F        40
## 5  34 full-time      F        24
## 6  61 part-time      F        21
```

Our main aim is to understand whether women earn less than men. First look at how `age` and `Ptstatus` are related with log(WeeklyPay):

```
Weeklypay.lm.1<-lm(log(WeeklyPay)~Ptstatus+age,data=Weekly_Pay)
display(Weeklypay.lm.1)
```

```
## lm(formula = log(WeeklyPay) ~ Ptstatus + age, data = Weekly_Pay)
##                   coef.est coef.se
## (Intercept)        5.91     0.03
## Ptstatuspart-time -0.68     0.01
## age                0.01     0.00
## ---
## n = 10218, k = 3
## residual sd = 0.73, R-Squared = 0.19
```

The coefficient of `Ptstatus` is significant and exp(-0.68)=0.51 meaning that people working part-time (under 20 hours p/w) earn almost 50% less than then full-time employed people a week at the same age.

Let's include `gender`

```
Weeklypay.lm.2<-lm(log(WeeklyPay)~gender+age+Ptstatus,data=Weekly_Pay)
display(Weeklypay.lm.2)
```

```
## lm(formula = log(WeeklyPay) ~ gender + age + Ptstatus, data = Weekly_Pay)
##                   coef.est coef.se
## (Intercept)        4.91     0.03
## genderM            1.03     0.02
## age                0.01     0.00
## Ptstatuspart-time -0.26     0.01
## ---
## n = 10218, k = 4
## residual sd = 0.61, R-Squared = 0.44
```

The coefficient of `Ptstatus` has changed to exp(-0.26)=0.77 meaning that part-time workers earn 23% less than full-time employees at the same age for women. However men earn exp(1.03)=2.8 almost 3 times as much as women at the same age if they are in full-time employment! That is a lot – it isn't that much in reality (usually men earn between 10-30% more than women on average). This is a doctored data-set.

We can see that if we hadn't included gender we might have drawn incorrect conclusions about the impact of being part-time on Weekly pay. `gender` and `Ptstatus` are clearly related as women tend to make up the larger part of the part-time work-force. As they are paid less than men even when in full-time employment

```
with(Weekly_Pay,table(Ptstatus,gender))
```

```
##            gender
## Ptstatus       F    M
##   full-time  380 4842
##   part-time 2412 2584
```

There are many situations in which you may want to consider whether there are variables that you should have in order to make any sense of the analysis. When looking at data about people (surveys, panel studies, cohort studies) you must always have things like age, gender, socio-economic status, education.

Creating a biased data set

Using `gender` and `age` from the Weekly pay dataset we create a new variable biased.pred which is the main dirver of newpay (a new outcome).

```
biased.rel<-rnorm(n=nrow(Weekly_Pay),mean=2,sd=2)
#creates a noisy "coefficient" of gender
biased.pred<-ifelse(Weekly_Pay$gender=="M",1,0)*biased.rel
# biased.pred is a noisy function of gender
biased.pred<-biased.pred-(Weekly_Pay$age-mean(Weekly_Pay$age))/(2*sd(Weekly_Pay$age))
# add to gender the normalised age
summary(biased.pred)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -6.39073 -0.04907  0.99676  1.43673  2.77520  9.83338
```

```
newpay<-with(Weekly_Pay,biased.pred-ifelse(gender=="M",1,0)*0.5+0.1*age+rnorm(nrow(Weekly_Pay),0,3))
#new pay is the biased predictor MINUS the gender and plus the age + some random error
summary(newpay)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -7.720   2.866   5.271   5.288   7.693  19.187
```

```
newpay.lm.1<-lm(newpay~gender+age,data=Weekly_Pay)
display(newpay.lm.1)
```

```
## lm(formula = newpay ~ gender + age, data = Weekly_Pay)
##             coef.est coef.se
## (Intercept) 1.61     0.14
## genderM     1.45     0.08
## age         0.06     0.00
## ---
## n = 10218, k = 3
## residual sd = 3.47, R-Squared = 0.07
```

```
newpay.lm.2<-lm(newpay~age+gender+biased.pred,data=Weekly_Pay)
display(newpay.lm.2)
```

```
## lm(formula = newpay ~ age + gender + biased.pred, data = Weekly_Pay)
##             coef.est coef.se
## (Intercept) -0.03     0.12
## age          0.10     0.00
```

```
## genderM     -0.56     0.08
## biased.pred  1.01     0.02
## ---
## n = 10218, k = 4
## residual sd = 3.01, R-Squared = 0.30
```

```
#very different coefficients for gender with and without the biased.pred.
```

Can you create a new variable that acts like this?