**Video Script: Section 1 Video 5 – Using ggplot in scripts**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Description | Action on screen | Narration |
| 1 | Introduction  (Outcome and why it is desirable)  This should give the viewer an idea of the outcome of the task at the beginning of the videos and set the stage and expectations of the viewer. | Opening slides. | **In this video**, we will see how to use ggplot2 in a script rather than in a console. |
| 2 | Context(Problem/Solution)  Present the viewer with a real-world solution and how the situation would pose as a challenge. It always helps to draw the viewer's attention using a use-case. Metadata template can be used here. |  | So far we used ggplot using one-liners in the console. This is obviously not sustainable if we want to make more complex graphs and use them in scripts. |
| 3 | Guidance (How to do it and how it works): | Open activity01\_05.R in the editor.  Highlight and run:  library(ggplot2)  set.seed(42)  small <- diamonds[sample(nrow(diamonds),1000),]  head(small) | Open activity\_01\_05.R in RStudio.  Run the first few lines to prepare some data. |
| 4 |  | Run the first line of example 01  p <- ggplot(small, aes(x = carat, y = price)) | As we have seen before, a plot is made up of different layers.  We can store the ggplot object in a variable (say 'p') and keep on adding to it.  Highlight the first line of example 01 and run it (CTRL+ENTER). |
| 5 |  |  | By running the first line of example 01, we create a ggplot object, define some data and the aesthetics x and y, and store the object in the variable p. |
| 6 |  | Run:  p + geom\_point() | We can now use this object for other plots:  Run the second line of example 01. |
| 7 |  | 01_05_blackwhite.png | You will get the black and white scatter plot that corresponds to the ggplot commands we set by adding the geom\_point layer to the object ‘p’. |
| 8 |  | Run:  p + geom\_point(aes(colour = cut))  01_05_colour.png | Likewise, run the third line of example 01.  The colourful scatter plot corresponds to the ggplot commands we set by adding the different geom\_point layer (with the extra ‘colour’ aesthetics mapped) to the object ‘p’. |
| 9 |  |  | By storing the ggplot object intoas a variable, we don't need to rely on one long line of commands but we can build very complex graphics one layer at the time. |
| 10 |  | Click on 'clear all'. All the plots disappear. | Now, clear all the plots by clicking 'clear all' In the ‘plots’ pane |
| 11 |  | Click on 'Source'. No plot appear. | Click on 'source' in the editor to run the code as a script, as opposed to running it in the console. |
| 12 |  |  | As you can see, none of the plots appear.  The reason the plots were actually produced and appearing when run in the console is that by typing the ggplot command, we implicitly printed the object, the same way you see the value of a variable when you enter its name in the console. |
| 13 |  | change  p + geom\_point()  to  print(p + geom\_point()) | To produce the plot from a script (and thus see it appearing on the screen when you click on 'source'), you need to make the call to print() explicit:  change  p + geom\_point()  to  print(p + geom\_point()) |
| 14 |  | click on 'save' and 'source'. A black and white plot appear.  01_05_blackwhite.png | Save and source the script. You now see the graph appearing on the screen.  In fact, the plot will only be built when a call to print() is made. You typically build complex graphs rapidly, and have the slower rendering process happening only when required. |
| 15 |  |  |  |
| 16 | Conclusion:The video concludes by showing the viewer that the goal has been achieved, and reminding them why they should be happy about that. A PowerPoint summary slide with the key points emphasized would make it easier for the viewer to remember what was covered in the video | Last slide of the PPT | We have seen how we can gradually build complex ggplot objects by storing them in a variable, rather than using long one-liners.  We also saw how to produce graphics from a script, rather than from the console.  In the next video, we will learn how to use geom\_line() to draw lines very easily. |