

Statistical Computing with R: Masters in Data Sciences 503 (S30) Fourth Batch, SMS, TU, 2025

Shital Bhandary

Associate Professor

Statistics/Bio-statistics, Demography and Public Health Informatics

Patan Academy of Health Sciences, Lalitpur, Nepal

Faculty, FAIMER Fellowship in Health Professions Education, India/USA

Review Preview:

- R script
- R markdown
- R notebook
 - Like jupyter notebook
- Project in R studio
- Working with Project
- Version control of Project
- R Shiny application

Chapter 8 (Workflow:projects) of the course text book: R for Data Science

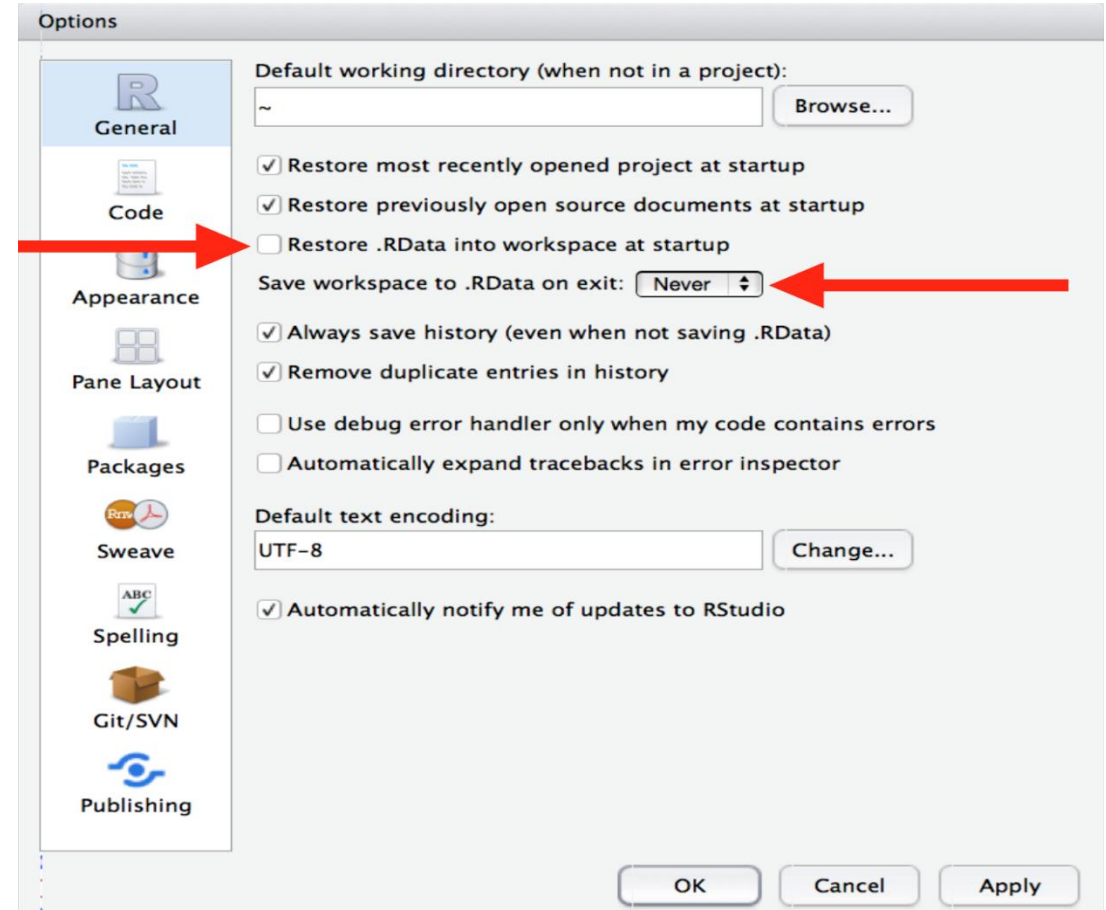
- One day you will need to quit R, go do something else and return to your analysis the next day.
- One day you will be working on multiple analyses simultaneously that all use R and you want to keep them separate.
- One day you will need to bring data from the outside world into R and send numerical results and figures from R back out into the world.
- To handle these real life situations, you need to make two decisions:
 - What about your analysis is “real”, i.e. what will you save as your lasting record of what happened?
 - Where does your analysis “live”?

Chapter 8 (Workflow:projects) of the course text book: R for Data Science

- As a beginning R user, it's OK to consider your environment (i.e. the objects listed in the environment pane) "real".
- However, in the long run, you'll be much better off if you consider your R scripts as "real".
- With your R scripts (and your data files), you can recreate the environment. It's much harder to recreate your R scripts from your environment!
- You'll either have to retype a lot of code from memory (making mistakes all the way) or you'll have to carefully mine your R history.

Chapter 8 (Workflow:projects) of the course text book: R for Data Science

- To foster this behaviour, it is highly recommended to instruct RStudio not to preserve your workspace between sessions.



Chapter 8 (Workflow:projects) of the course text book: R for Data Science

- This will cause you some short-term pain, because now when you restart RStudio it will not remember the results of the code that you ran last time.
- But this short-term pain will save you long-term agony because it forces you to capture all important interactions in your code.
- There's nothing worse than discovering three months after the fact that you've only stored the results of an important calculation in your workspace, not the calculation itself in your code.
- Therefore we need "Project" and R Studio provides this feature to us!

R Studio Project:

<https://support.rstudio.com/hc/en-us/articles/200526207-Using-RStudio-Projects>

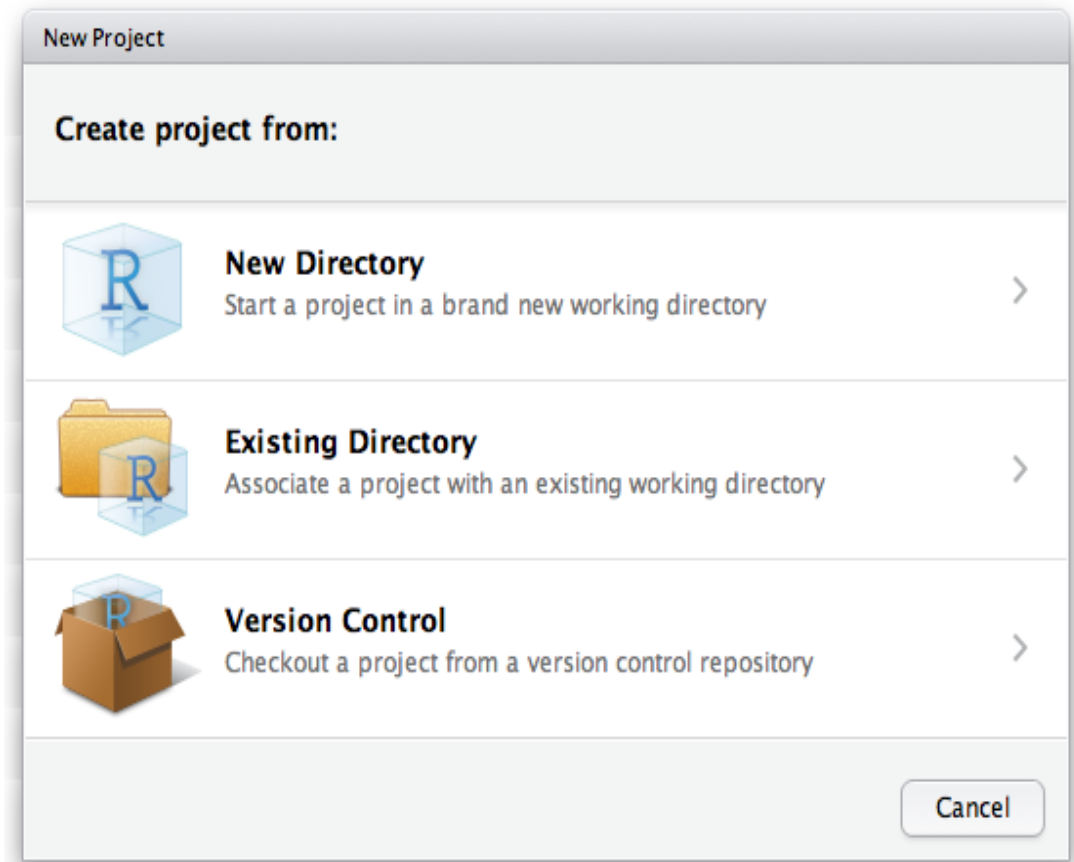
- R experts keep all the files associated with a project together — input data, R scripts, analytical results, figures.
- This is such a wise and common practice that RStudio has built-in support for this via **projects**.
- RStudio projects make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents.

#Creating Projects

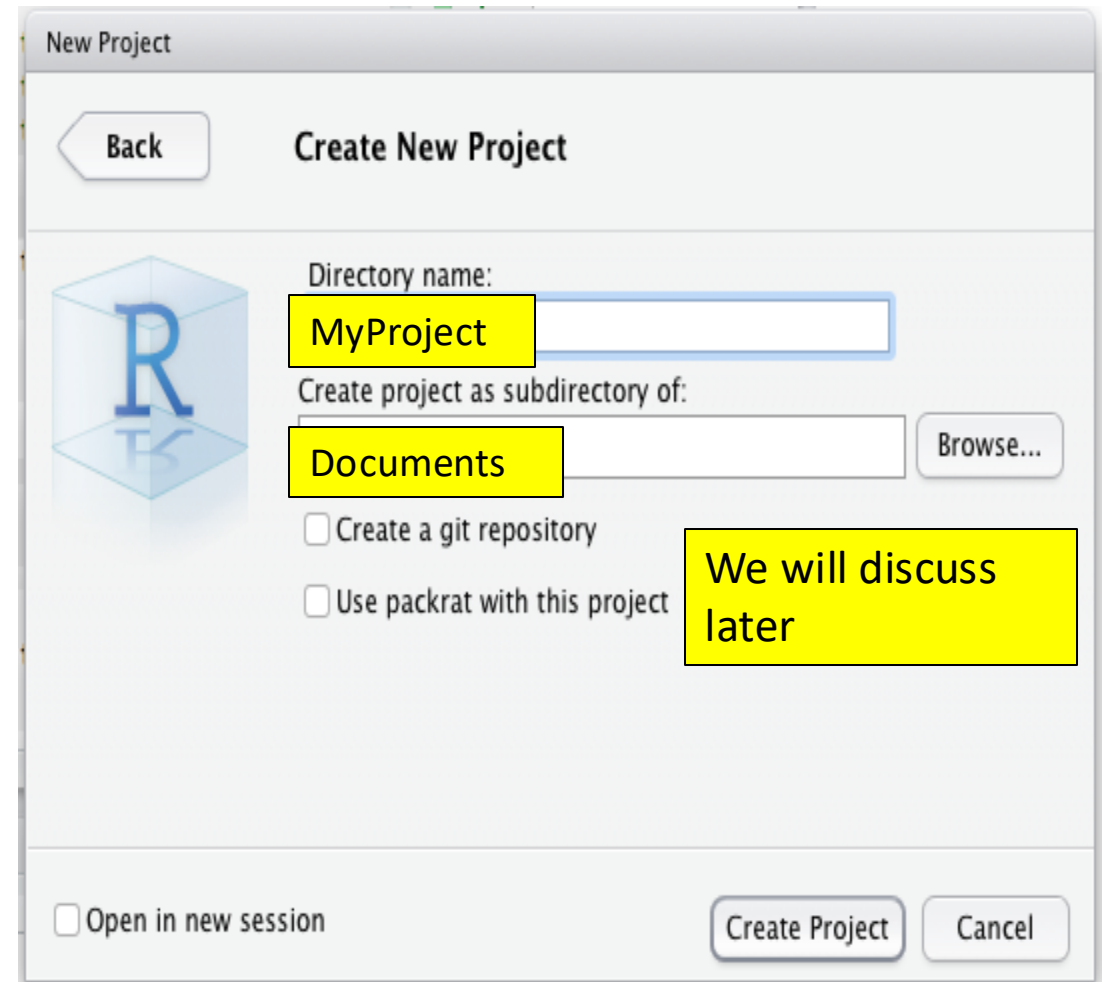
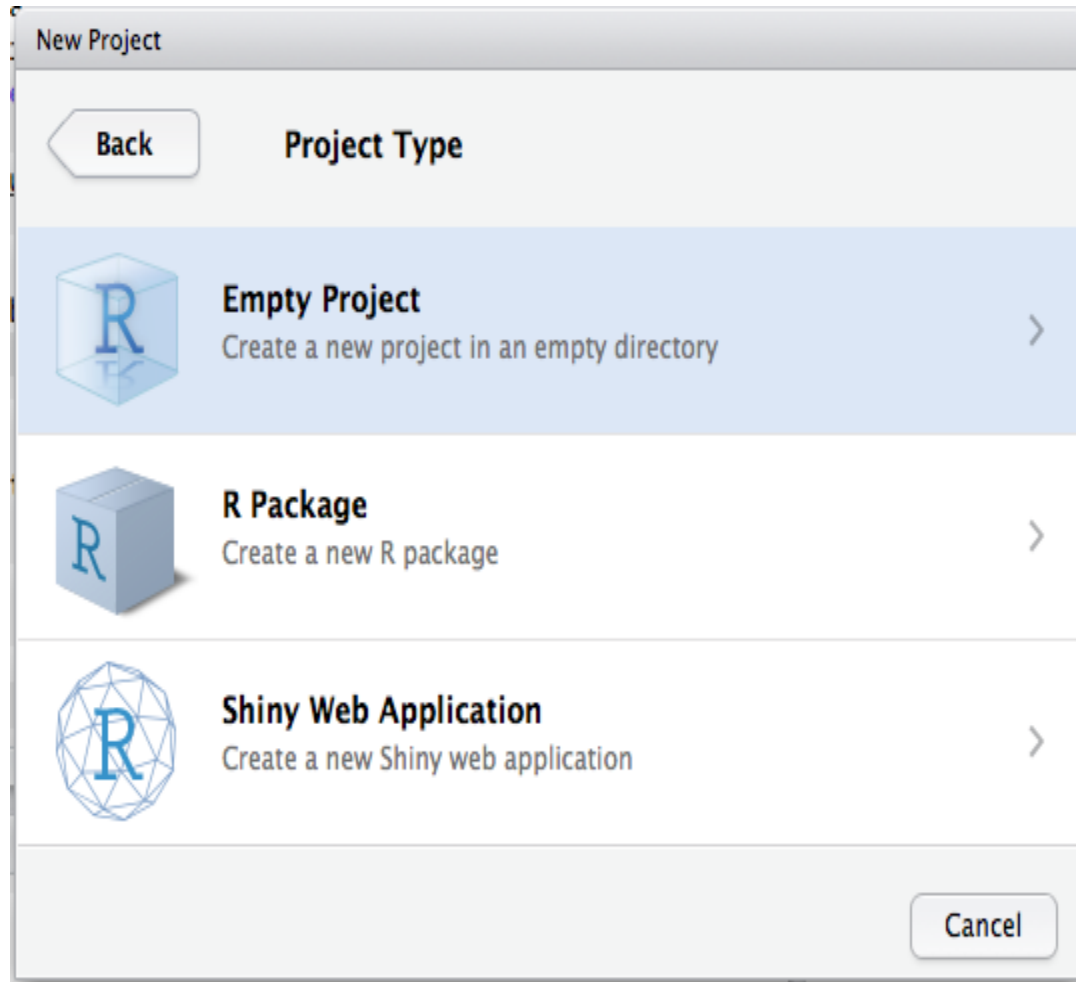
- RStudio projects are associated with R working directories.
- You can create an RStudio project:
 - In a brand new directory
 - In an existing directory where you already have R code and data
 - By cloning a version control (Git or Subversion) repository

R Studio project: Let's do it

- To create a new project in the RStudio IDE, use the **Create Project** command (available on the Projects menu and on the global toolbar):



Working with Project: Let's do it



Working with Project:

- When a new project is created RStudio:
- Creates a project file (**with an .Rproj extension e.g. MyProject.Rproj**) within the project directory.
- This file contains various project options (discussed below) and can also be used as a shortcut for opening the project directly from the file system.
- Creates a **hidden directory** (named **.Rproj.user**) where project-specific temporary files (e.g. auto-saved source documents, window-state, etc.) are stored. This directory is also automatically added to **.Rbuildignore**, **.gitignore**, etc. if required.
- Loads the project into RStudio and display its name in the Projects toolbar (**which is located on the far right side of the main toolbar**)

Working with Project:

- Call your project “MyProject” and think carefully about which *subdirectory* you put the project in.
- If you don’t store it somewhere sensible, it will be hard to find it in the future!
- Once this process is complete, you’ll get a new RStudio project.
- Check that the “home” directory of your project is the current working directory with:
 - `getwd()`
 - "C:/Users/Dell/Documents/MyProject"

Working with Project:

- Now **enter the following commands** in the script editor, and **save the file**, calling it “diamonds.R” **inside/in the project directory** →
- Next, **run the complete script** which will save a PDF and CSV file into **your project directory**.
- Quit RStudio & Inspect the folder associated with your project — notice the MyProject.**Rproj** file!

```
#diamonds.R
```

```
library(tidyverse)
ggplot(diamonds, aes(carat, price)) +
  geom_hex()
ggsave("diamonds.pdf")
write_csv(diamonds, "diamonds.csv")
```

I got this in Dell Optiplex 3010: Windows 10

MyProject - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Package tidyverse required but is not installed. [Install](#) [Don't Show Again](#)

```
1 #diamonds.R
2
3 library(tidyverse)
4 ggplot(diamonds, aes(carat, price)) +
5   geom_hex()
6 ggsave("diamonds.pdf")
7 write_csv(diamonds, "diamonds.csv")
8
```

tidyverse was installed a priori but it is asking to install for this project!
When I clicked Install, it says “installing diamond.R” dependencies!

Environment History Connections Tutorial

Global Environment

Environment is empty

Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > MyProject

	Name	Size	Modified
	..		
	MyProject.Rproj	218 B	Jan 10, 2022, 11:29 AM
	diamonds.R	150 B	Jan 10, 2022, 11:30 AM

Console Terminal Jobs

~/MyProject/

R is free software and comes with absolutely no warranty.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |

ggsave asked me to install “hexbin” package as it was required for the geom_hex layer although it was also installed already!

Finally, I got this:

The screenshot displays the RStudio interface for a project named 'MyProject'. The main editor window shows a script named 'diamonds.R' with the following code:

```
1 #diamonds.R
2
3 library(tidyverse)
4 ggplot(diamonds, aes(carat, price)) +
5   geom_hex()
6 ggsave("diamonds.pdf")
7 write_csv(diamonds, "diamonds.csv")
8
```

The Environment pane on the right shows the 'Global Environment' with the message 'Environment is empty'. The Files pane at the bottom right shows the project structure:

Name	Size	Modified
..		
MyProject.Rproj	218 B	Jan 10, 2022, 11:29 AM
diamonds.R	150 B	Jan 10, 2022, 11:30 AM
diamonds.csv	2.3 MB	Jan 10, 2022, 11:34 AM
diamonds.pdf	12.5 KB	Jan 10, 2022, 11:44 AM

The Console pane at the bottom shows the output of the script execution:

```
~/MyProject/
Selection: 1
Installing package into 'C:/Users/Dell/Documents/R/win-library/4.0'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.0/hexbin_1.28.2.zip'
content type 'application/zip' length 1474974 bytes (1.4 MB)
downloaded 1.4 MB

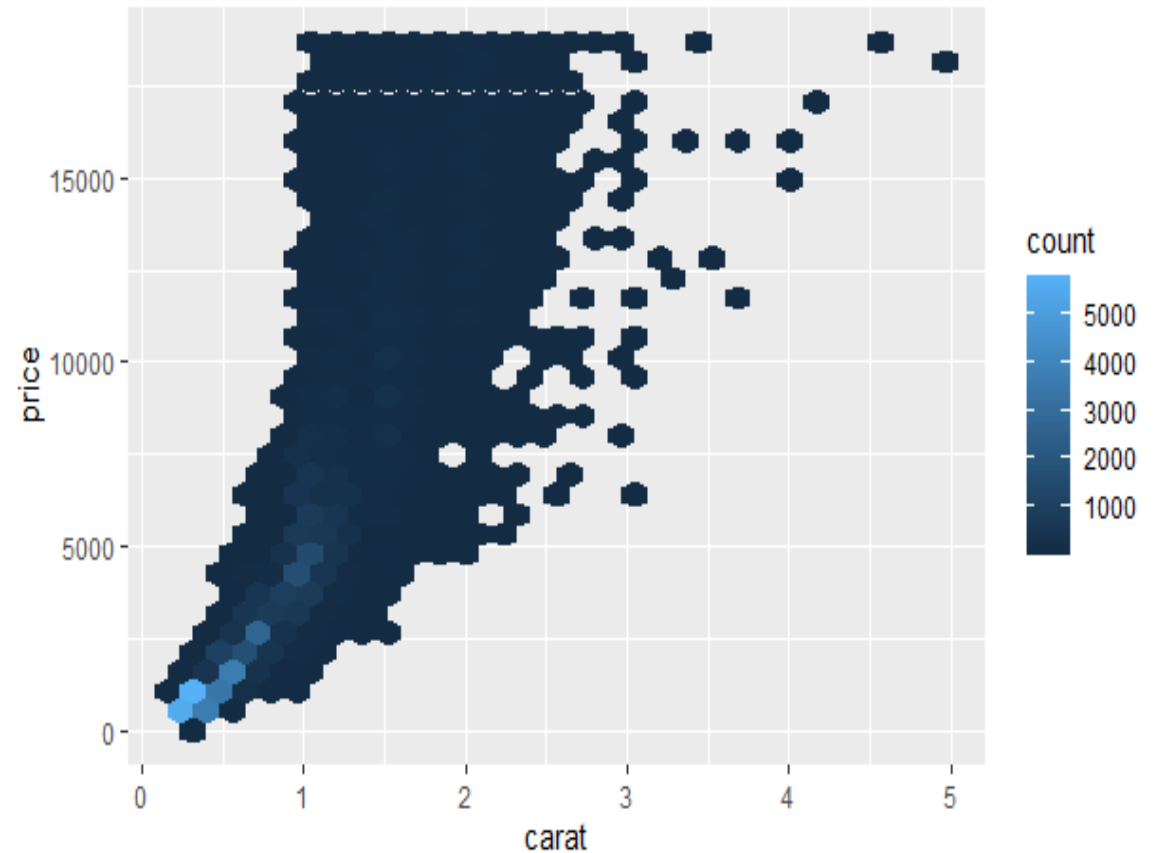
package 'hexbin' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:/Users/Dell/AppData/Local/Temp/RtmpkbnKiv/downloaded_packages
> write_csv(diamonds, "diamonds.csv")
> ggsave("diamonds.pdf")
Saving 6.4 x 3.78 in image
>
```

The plot: Saved as PDF and also seen in the “plots” tab

#It was produced due to this code:

- `library(tidyverse)`
- `ggplot(diamonds, aes(carat, price)) +`
- `geom_hex()`



Working with Project:

Opening Projects

- There are several ways to open a project:
- Using the **Open Project** command (available from both the Projects menu and the Projects toolbar) to browse for and select an existing project file
 - e.g. MyProject.Rproj
- Selecting a project from the list of most recently opened projects (also available from both the Projects menu and toolbar).
- Double-clicking on the project file within the system shell (e.g. Windows Explorer, OSX Finder, etc.).

Working with Project:

- When a project is opened within RStudio the following actions are taken:
- A new R session (process) is started
- The **.Rprofile** file in the project's main directory (if any) **is sourced by R**
- The **.RData** file in the project's main directory **is loaded** (if project options indicate that it should be loaded).
- The **.Rhistory** file in the project's **main directory is loaded** into the RStudio History pane (and used for Console Up/Down arrow command history).
- The **current working directory is set to the project directory.**
- Previously edited source documents are restored into editor tabs
- Other **RStudio settings** (e.g. active tabs, splitter positions, etc.) **are restored to where they were the last time the project was closed.**

Working with Project:

Quitting a Project

- When you are within a project and choose to either Quit, close the project, or open another project the following actions are taken:
- **.RData and/or .Rhistory are written to the project directory** (if current options indicate they should be)
- The list of open source documents is saved (so it can be restored next time the project is opened)
- Other RStudio settings (as described above) are saved.
- The R session is terminated.

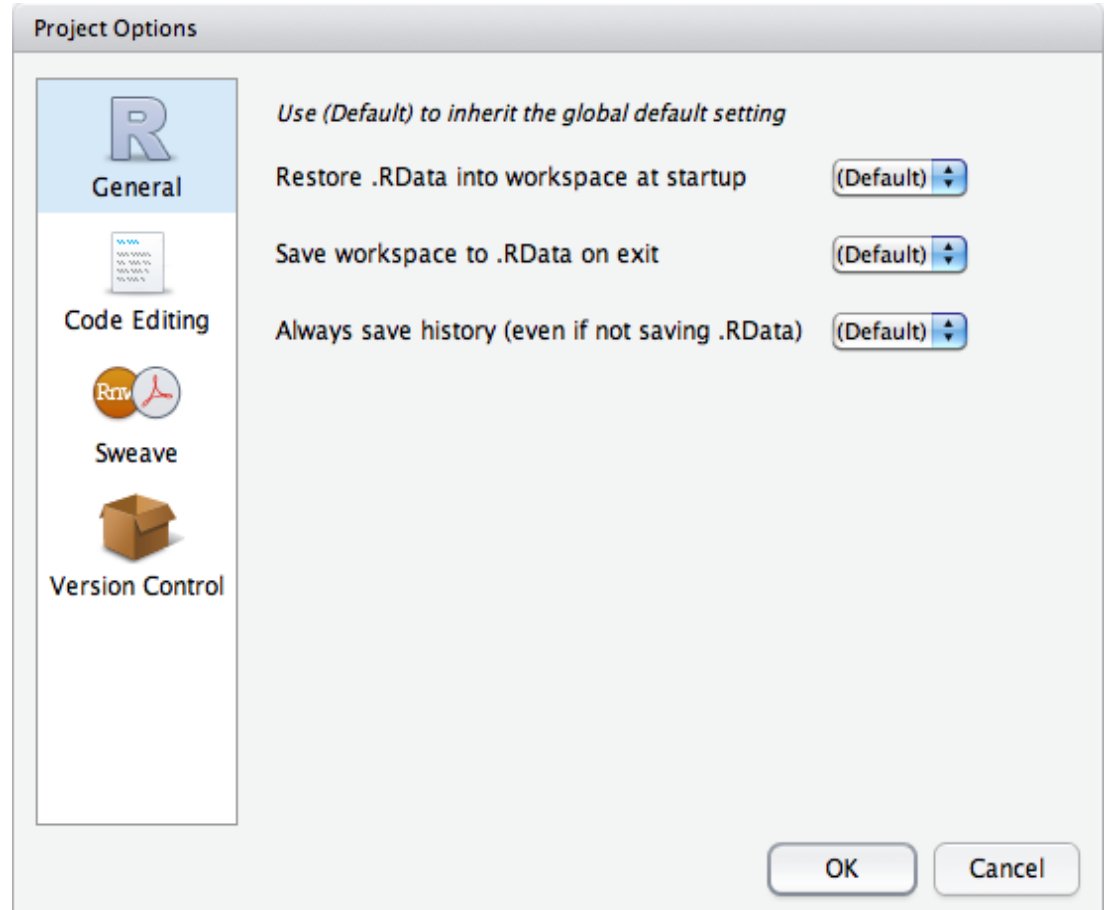
Working with Project:

Working with Multiple Projects at Once

- You can work with more than one RStudio project at a time by simply opening each project in its own instance of RStudio.
- There are two ways to accomplish this:
 - Use the **Open Project in New Window** command located on the Project menu.
 - Opening **multiple project files via the system shell** (i.e. double-clicking on the project file).

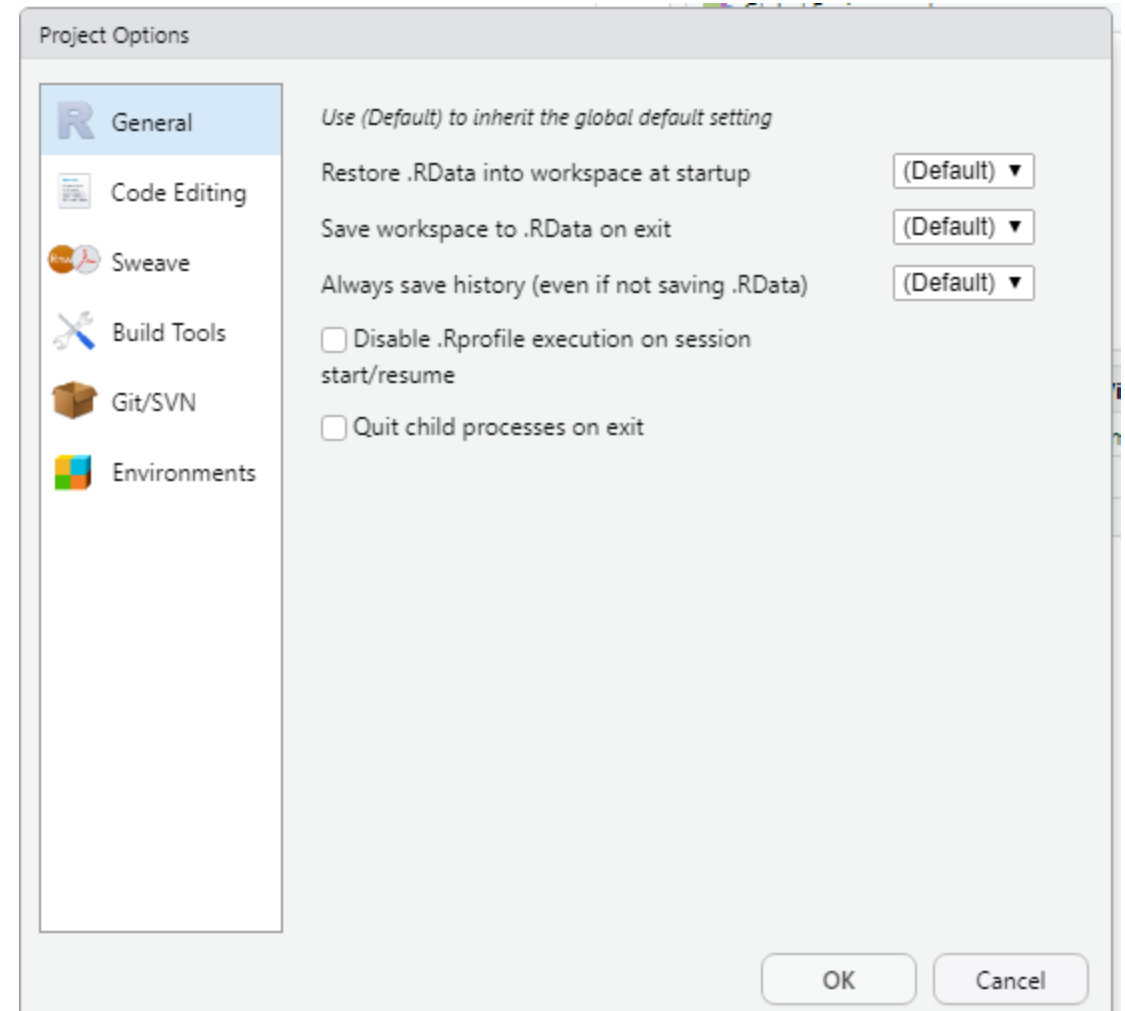
Working with Project:

- Tools → Project Options
- There are several options that can be set on a per-project basis to customize the behavior of RStudio. You can edit these options using the **Project Options** command on the **Project** menu:



You can explore each of the options seen:

- **General** (*Do not change!*)
- Code Editing
- Sweave
- Build Tools
- Git/SVN
- Environments
 - packrat would appear if it was selected while creating the project



Working with Project:

- Example: Code Editing
- **Index R source files** — Determines whether R source files within the project directory are indexed for code navigation (i.e. go to file/function, go to function definition).
- Normally this should remain enabled, however if you have a project directory with thousands of files and are concerned about the overhead of monitoring and indexing them you can disable indexing here.
- **Insert spaces for tab** — Determine whether the tab key inserts multiple spaces rather than a tab character (soft tabs). Configure the number of spaces per soft-tab.
- **Text encoding** — Specify the default text encoding for source files.
- Note that source files which don't match the default encoding can still be opened correctly using the **File : Reopen with Encoding** menu command.

Working with Project:

#Project options: Sweave

PDF Generation

- Weave Rnw (or Sweave) file using: Sweave
- Typeset LaTeX into PDF using: pdfLaTeX

PDF Preview

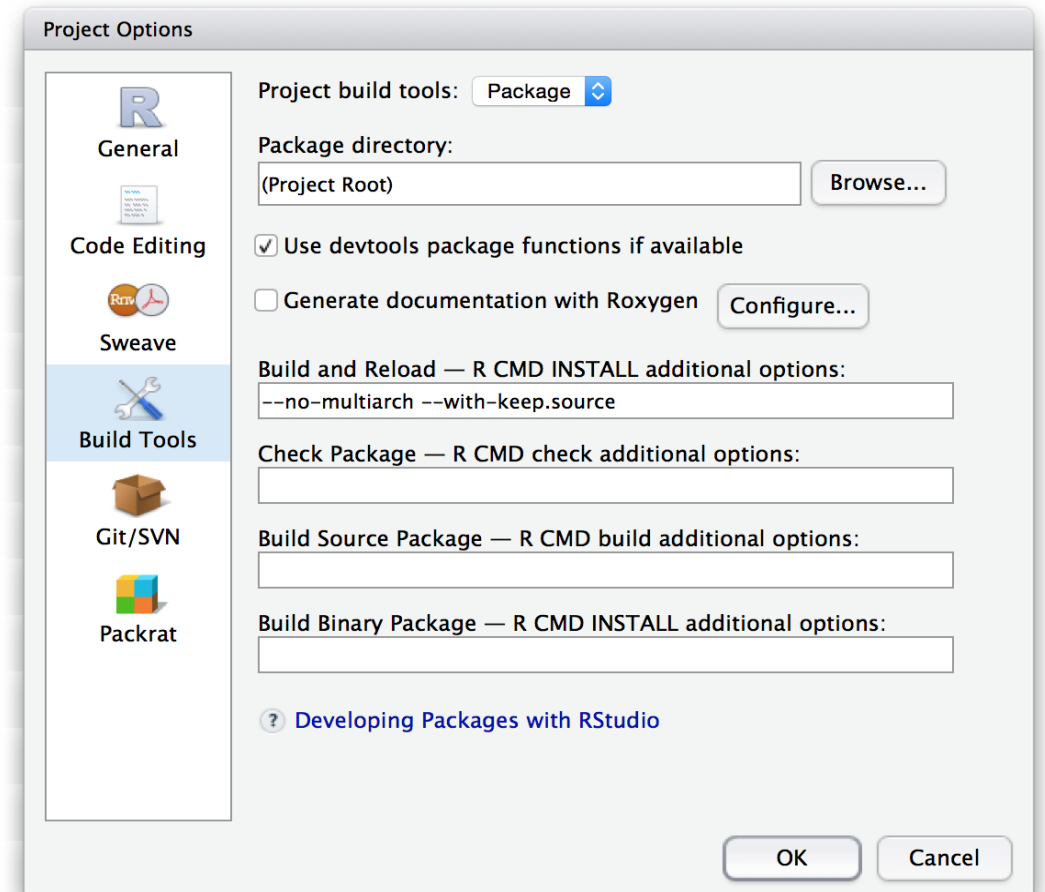
- Compile PDF root document:

- More here:
<https://support.rstudio.com/hc/en-us/articles/200532257?version=1.3.1093&mode=desktop>
- <https://support.rstudio.com/hc/en-us/articles/200532247?version=1.3.1093&mode=desktop>
- <https://support.rstudio.com/hc/en-us/articles/200486298?version=1.3.1093&mode=desktop>

Working with Project:

#Project options: Build Tools

- There are three R package build commands used by the package development tools in the RStudio IDE:
 - R CMD check
 - R CMD build
 - R CMD INSTALL
- It's possible to customize the options passed to each of these commands using **Project Options : Build Tools**



Working with Project:

- **Version control system** - Specify the version control system to use with this project.
- Note that **RStudio automatically detects** the presence of version control for projects by scanning for a .git or .svn directory.
- **Therefore it isn't normally necessary to change this setting.**
- You may want to change the setting for the following reasons:
 - You have both a .git and .svn directory within the project and wish to specify **which version control system RStudio should bind to.**
 - You have no version control setup for the project and you **want to add a local git repository** (equivalent to executing git init from project root directory)
- **Origin** — Read-only display of the remote origin (if any) for the project version control repository

More here: <https://support.rstudio.com/hc/en-us/articles/200532077?version=1.3.1093&mode=desktop>

Github account and a repository

The screenshot shows the GitHub interface for a repository named 'covid19_nepal' by user 'shitalbhandary'. The repository is public and has 0 stars, 1 watch, and 0 forks. The 'Code' button is highlighted in green, and its dropdown menu is open, showing options to clone the repository using HTTPS, SSH, or GitHub CLI, or to open it with GitHub Desktop or download it as a ZIP file. The repository's file list is visible on the left, and the 'About' section on the right provides more details about the project.

Repository: shitalbhandary / covid19_nepal (Public)

Navigation: Code (selected), Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, Settings

Branches and Tags: main (1 branch), 0 tags

Files and Folders:

File/Folder	Status
code	Updated
data	Data and code
figures	Updated
.gitignore	Initial commit
LICENSE	Initial commit
README.md	Initial commit
covid19_nepal.Rproj	Getting project organized by f

Clone Options:

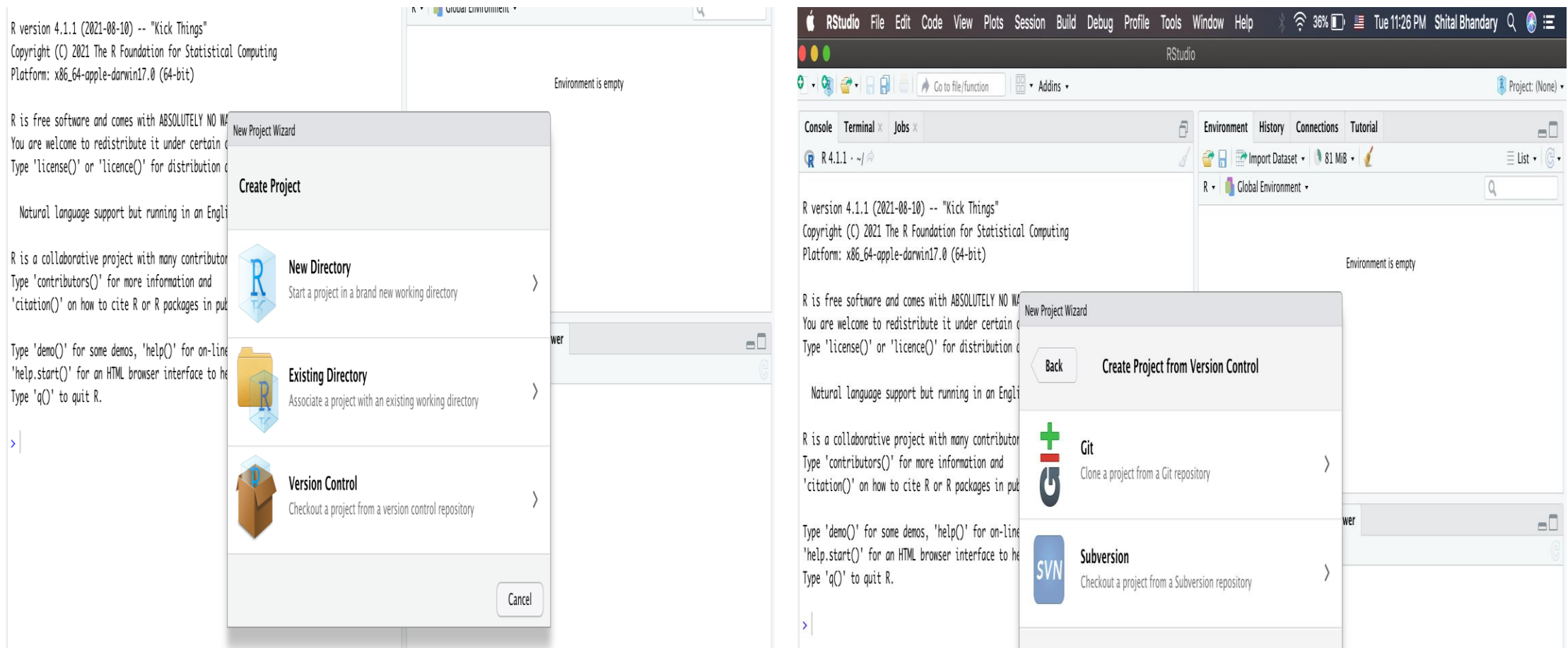
- Local
- Codespaces
- Clone (icon)
- HTTPS (selected)
- SSH
- GitHub CLI
- URL: <https://github.com/shitalbhandary/covid19>
- Use Git or checkout with SVN using the web URL.
- Open with GitHub Desktop
- Download ZIP

About: Wikipedia Covid 19 Data Analysis

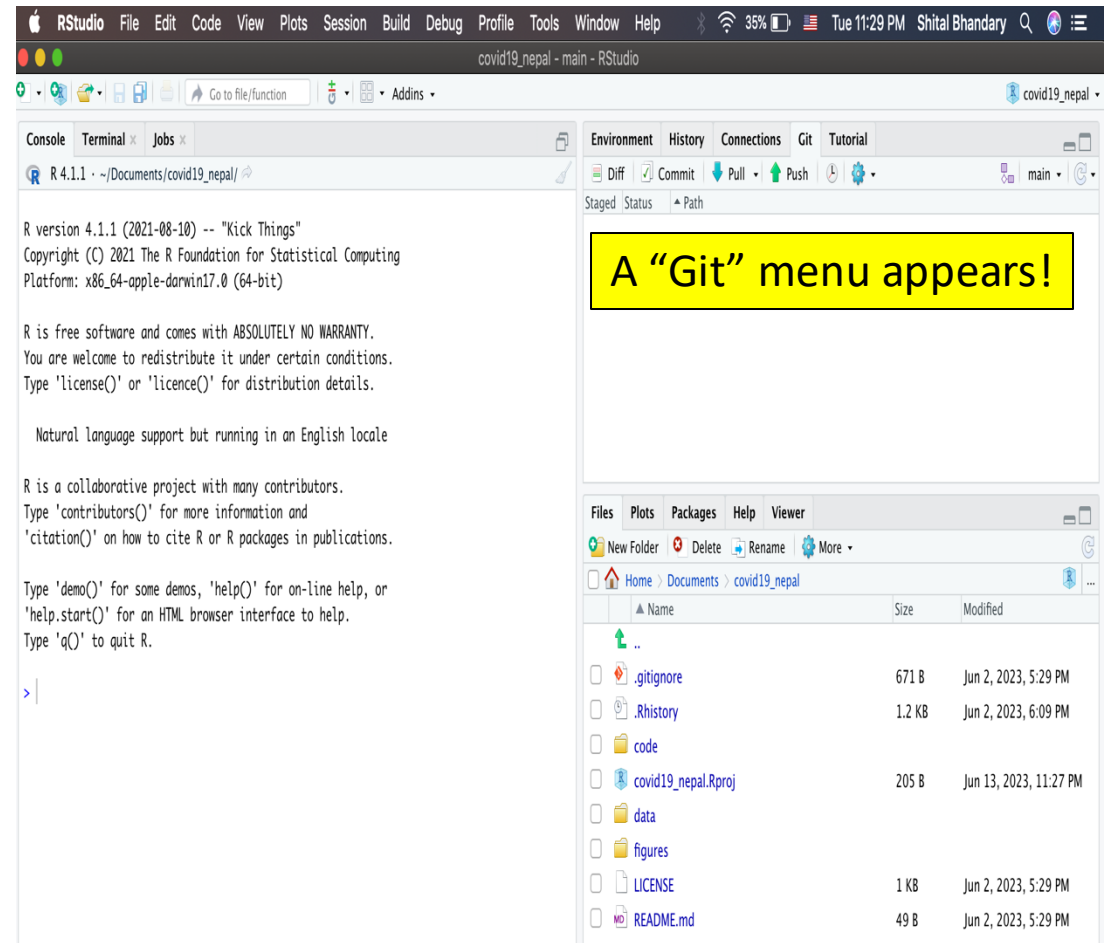
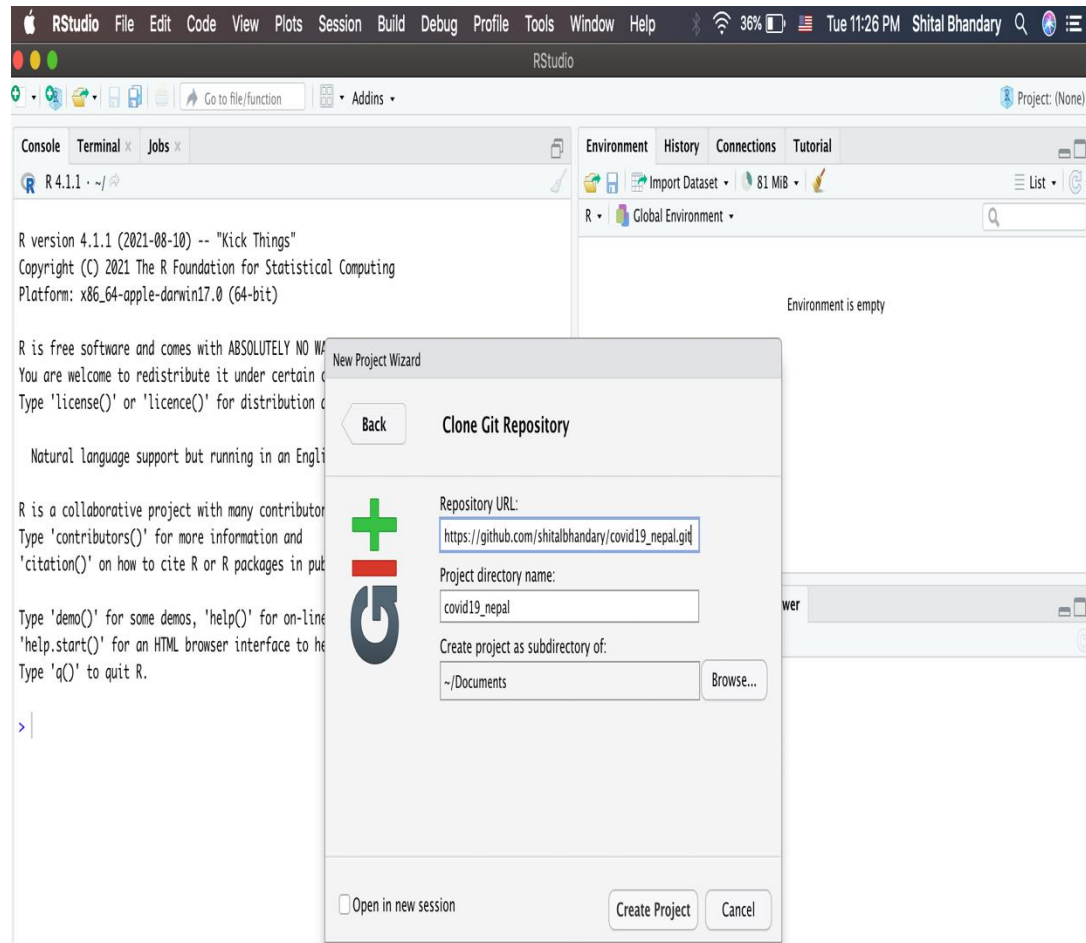
- Readme
- MIT license
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases: No releases published. [Create a new release](#)

R Studio and Github: Create new project → Version Control

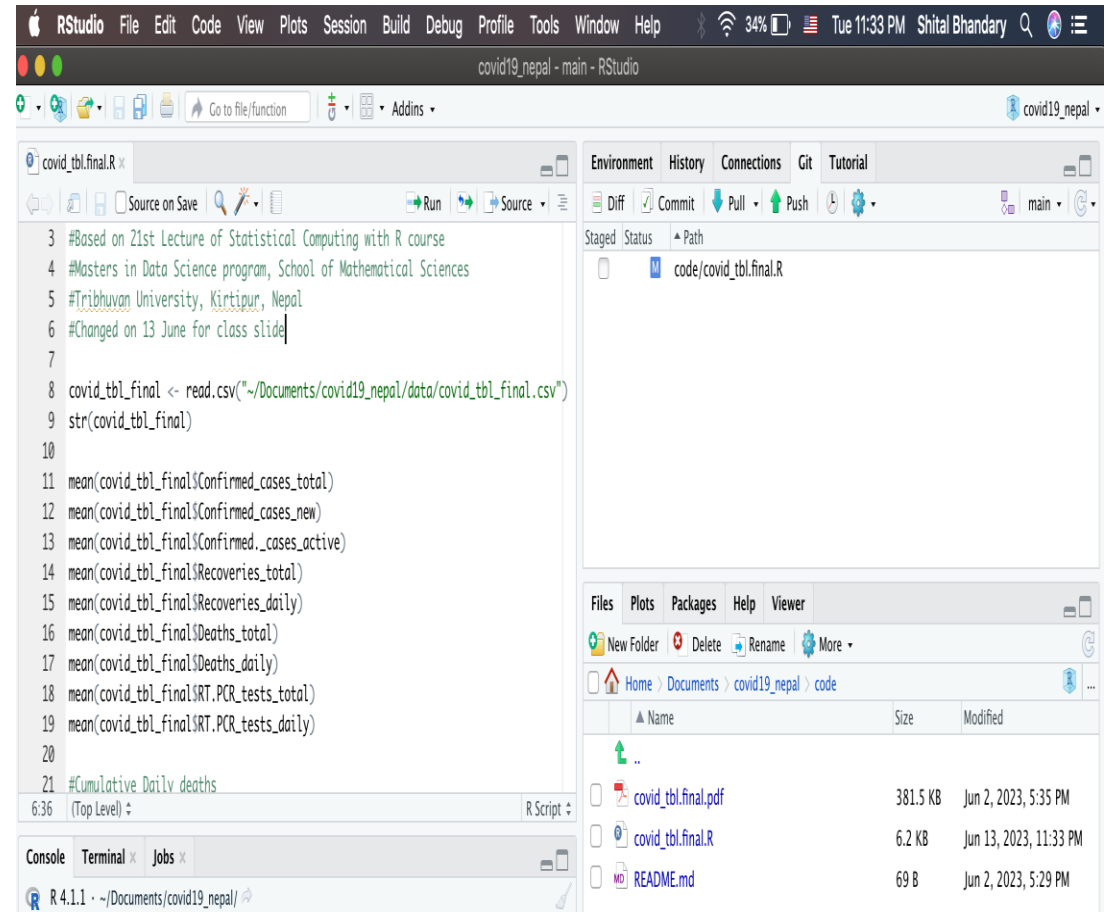


R Studio and Github: Git → Copied link of the github repository from the github web

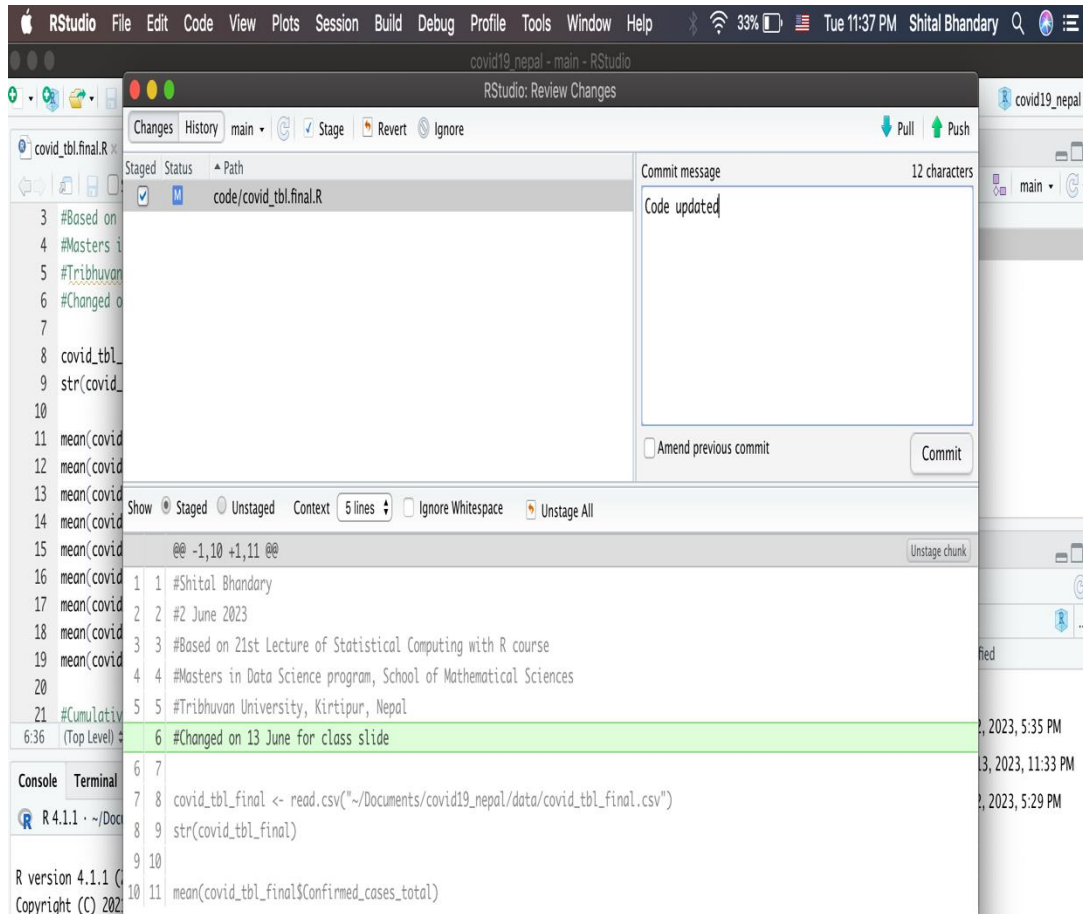


We can work with this repository in R Studio:

- When we make changes and hit save button then the changed file appears e.g. changes in the code/covid_tbl.final.R file is shown here
- We need to check it and hit the “commit” button and then type a commit message as a simple text e.g. “code updated”



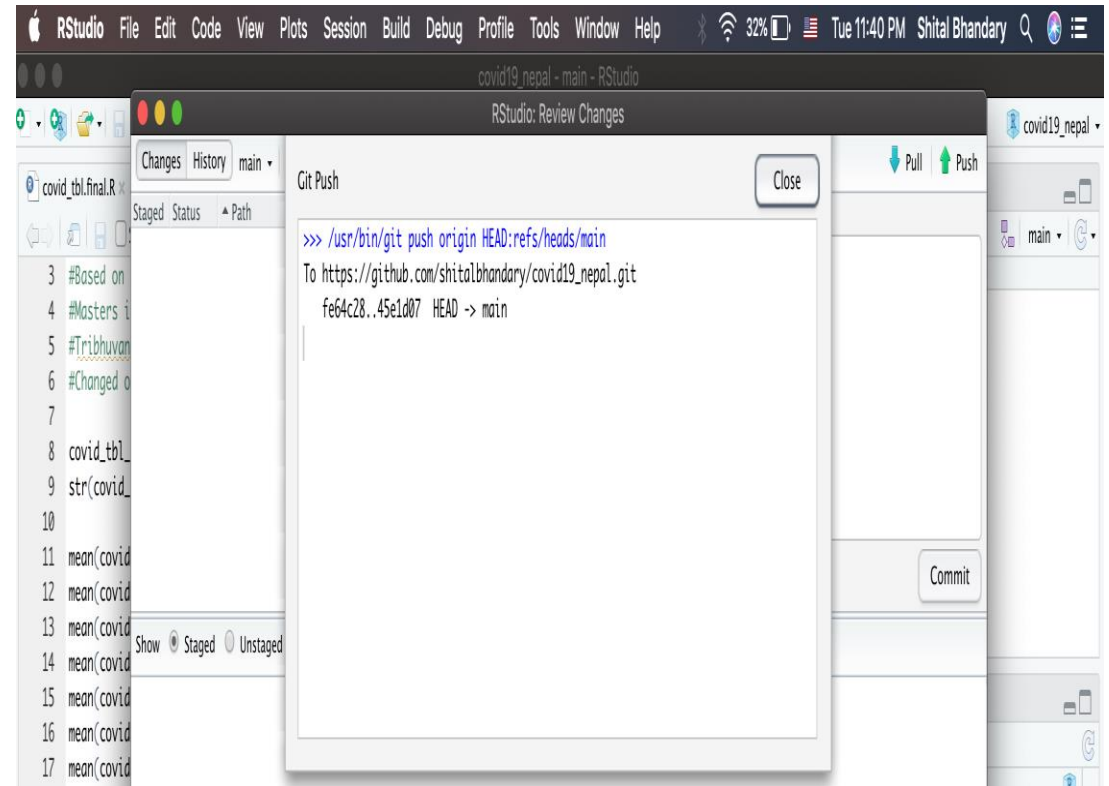
Once the “commit” is done then we need to “push” it to the github repo to save the changes!



The screenshot shows the RStudio Review Changes window. The 'Commit message' field contains the text 'Code updated'. The 'Commit' button is visible at the bottom right of the message field. The background shows the R script editor with a file named 'covid19_nepal.R' and a console window at the bottom.

```
##Based on
##Masters i
##Tribhuvan
##Changed o
7
8 covid_tbl
9 str(covid
10
11 mean(covid
12 mean(covid
13 mean(covid
14 mean(covid
15 mean(covid
16 mean(covid
17 mean(covid
18 mean(covid
19 mean(covid
20
21 #Cumulativ
22 (Top Level)
6:36

Console
R 4.1.1 - ~/Doc
R version 4.1.1 (
Copyright (C) 202
```



The screenshot shows the RStudio Review Changes window with a 'Git Push' dialog box open. The dialog box contains the command `>>> /usr/bin/git push origin HEAD:refs/heads/main` and the URL `To https://github.com/shitalbhandary/covid19_nepal.git`. The 'Commit' button is visible at the bottom right of the dialog box. The background shows the R script editor with a file named 'covid19_nepal.R' and a console window at the bottom.

```
3 ##Based on
4 ##Masters i
5 ##Tribhuvan
6 ##Changed o
7
8 covid_tbl
9 str(covid
10
11 mean(covid
12 mean(covid
13 mean(covid
14 mean(covid
15 mean(covid
16 mean(covid
17 mean(covid
```

This type of message will be seen if all goes well!

Working with Project:

- Project options: Environments
- Rstudio uses the “renv” package to give your projects their own privately managed library, making your R code more isolated, portable and reproducible
- Click/check: “Use “renv” with this project” to use this feature!

More here:

<https://rstudio.github.io/renv/>

#How to use:

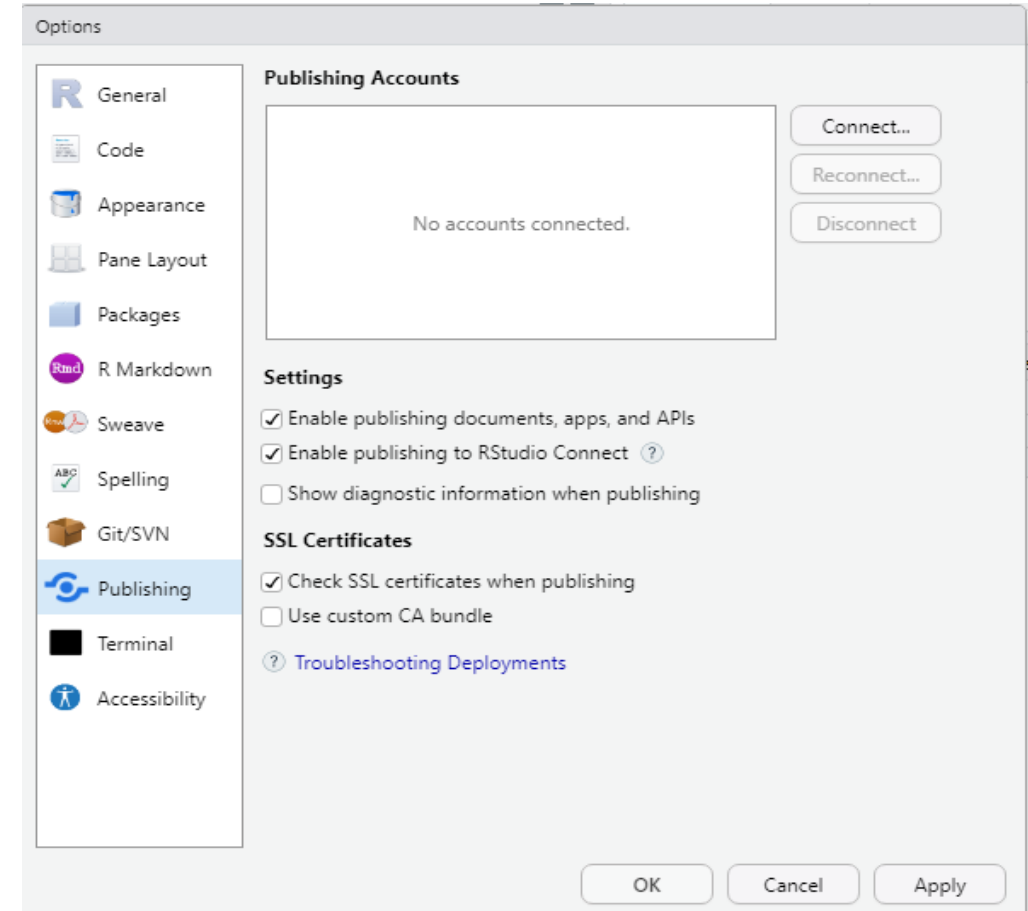
- Use **renv::init()** to initialize renv with a new or existing project.
- This will set up your project with a private library, and also make sure to install all of the packages you’re using into that library.
- The packages used in your project will be recorded into a *lockfile*, called `renv.lock`

Alternatively, we can use “packrat” too (define while creating project): <https://rstudio.github.io/packrat/rstudio.html>

Working with Project: Publishing your work!

- Publishing with R studio:
- Markdown/R notebook files using:
- **Rpubs (free!)**
- R Studio connect

More here: <https://support.rstudio.com/hc/en-us/articles/228270928-Push-button-publishing-to-RStudio-Connect>



More here: <https://docs.rstudio.com/connect/user/publishing/>

R Shiny web app: <https://shiny.posit.co/>

- We need to define
 - ui
 - server
 - shinyApp(ui, server)

- to get an interactive web app of data science projects for “free”

Example

- app.R (R Script)
 - `library(shiny)`
 - `library(bslib)`
 - `library(dplyr)`
 - `library(ggplot2)`

Getting the data and defining user interface (ui)

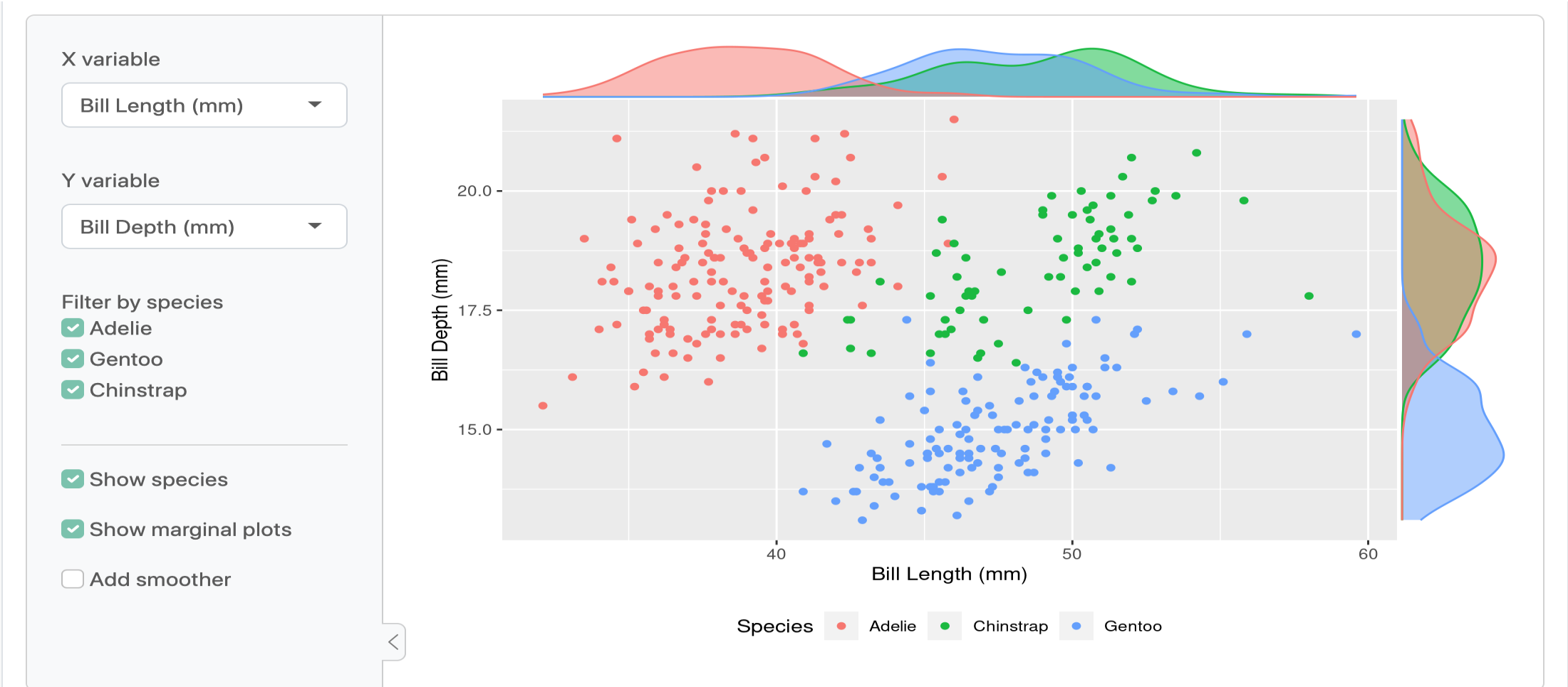
Data: <https://raw.githubusercontent.com/jcheng5/simplepenguins.R/main/penguins.csv>

- ```
ui <- page_fillable(theme = bs_theme(bootswatch = "minty"), layout_sidebar(fillable = TRUE, sidebar(varSelectInput("xvar", "X variable", df_num, selected = "Bill Length (mm)"), varSelectInput("yvar", "Y variable", df_num, selected = "Bill Depth (mm)"),
```
- ```
checkboxGroupInput("species", "Filter by species", choices = unique(df$Species), selected = unique(df$Species) ), hr(), # Add a horizontal rule checkboxInput("by_species", "Show species", TRUE), checkboxInput("show_margins", "Show marginal plots", TRUE), checkboxInput("smooth", "Add smoother"), ) plotOutput("scatter") ) )
```

R Shiny app server

- ```
server <- function(input, output, session) { subsetted <- reactive({ req(input$species) df |> filter(Species %in% input$species) })
```
- ```
output$scatter <- renderPlot({ p <- ggplot(subsetted(), aes(!!input$xvar, !!input$yvar)) + list(theme(legend.position = "bottom"), if (input$by_species) aes(color=Species), geom_point(), if (input$smooth) geom_smooth() )
```
- ```
if (input$show_margins) { margin_type <- if (input$by_species) "density" else "histogram" p <- p |> ggExtra::ggMarginal(type = margin_type, margins = "both", size = 8, groupColour = input$by_species, groupFill = input$by_species) } p }, res = 100) }
```
- ```
shinyApp(ui, server)
```

Output: We can put our Shiny app on the web by using our own servers or posit's hosting service!



Hosting and Deployment:

[Get Started](#)[Articles](#)[Gallery](#)[Reference](#)[Help](#)[Deploy](#)[Contribute](#)

Hosting and deployment

When it's time to put your Shiny app on the web, you can choose to deploy on your own servers or on our hosting service.

Deploy to the cloud

Shinyapps.io

Host your Shiny apps on the web in minutes with Shinyapps.io. It is easy to use, secure, and scalable. No hardware, installation, or annual purchase contract required. Free and paid options available.

[Learn more](#)[Get started](#)[FAQ](#)

Hosting and Deployment Pricing at Shinyapps.io website!

shinyapps.io by Posit

[Home](#)[Features](#)[Pricing](#)[Support](#)[Log In](#)

FREE

\$0 /month

New to Shiny? Deploy your applications for FREE.

5 Applications

25 Active Hours

✓ Community Support

STARTER

\$13 /month
(or \$145/year)

More applications. More active hours!

25 Applications

100 Active Hours

✓ Premium Email Support

BASIC

\$49 /month
(or \$550/year)

Take your users to the next level!

Unlimited Applications

500 Active Hours

✓ Performance Boost

✓ Premium Email Support

STANDARD

\$119 /month
(or \$1,330/year)

Password protection? Authenticate your users!

Unlimited Applications

2,000 Active Hours

✓ Authentication

✓ Performance Boost

✓ Premium Email Support

PROFESSIONAL

\$349 /month
(or \$3,860/year)

Professional has it all! Personalize your domains.

Unlimited Applications

10,000 Active Hours

✓ Authentication

✓ Account Sharing

✓ Performance Boost

Question/queries?

- Second Assessment

- Final Assessment

Thank you!

@shitalbhandary