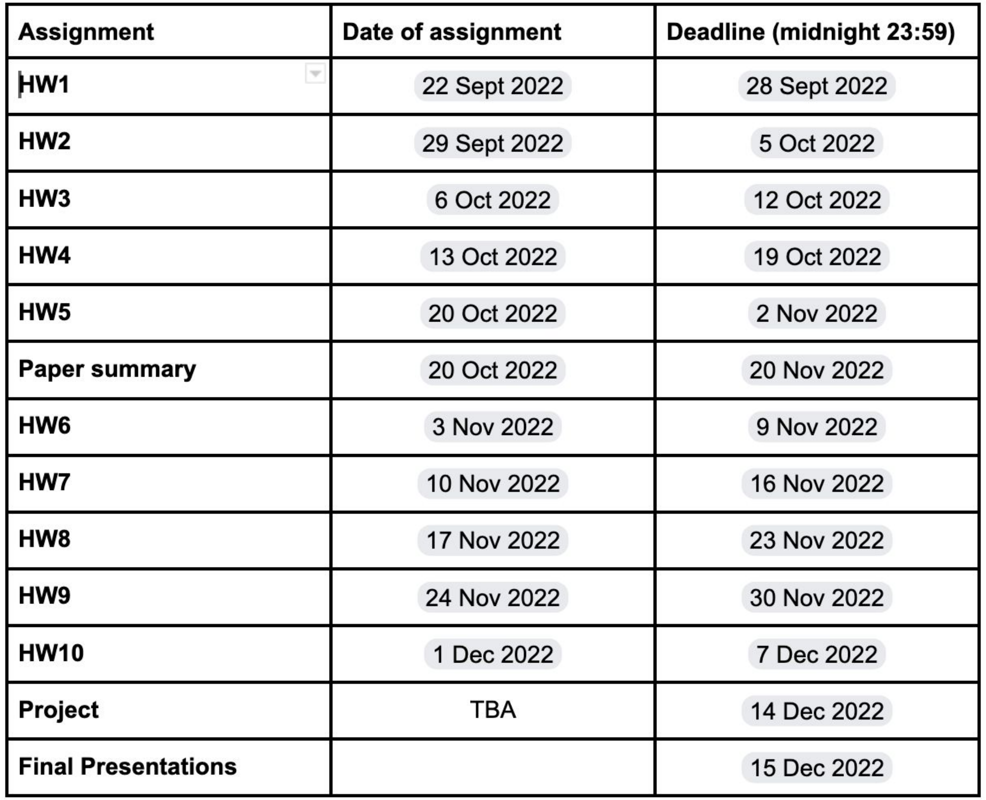
Data Analytics in R
Session 3

Maria Kunevich

Course information: assignments deadlines



Assignment	Date of assignment	Deadline (midnight 23:59)
HW1	22 Sept 2022	28 Sept 2022
HW2	29 Sept 2022	5 Oct 2022
HW3	6 Oct 2022	12 Oct 2022
HW4	13 Oct 2022	19 Oct 2022
HW5	20 Oct 2022	2 Nov 2022
Paper summary	20 Oct 2022	20 Nov 2022
HW6	3 Nov 2022	9 Nov 2022
HW7	10 Nov 2022	16 Nov 2022
HW8	17 Nov 2022	23 Nov 2022
HW9	24 Nov 2022	30 Nov 2022
HW10	1 Dec 2022	7 Dec 2022
Project	TBA	14 Dec 2022
Final Presentations		15 Dec 2022

Course information: homework assignments

- The deadlines for all assignments are **strict deadlines** (usually Wednesday 23:59)
- Penalty is **-20%** from the **max grade** for the assignment for **each extra** late day
- Generally, if you submit **after 23:59**, it is considered **1 day over** the deadline
- But you have **5 late days** to submit without penalty, you can use these days at your own will across all the assignments
- For example, you are late for an assignment:

=> penalty of -20% for each day you're late is applied automatically

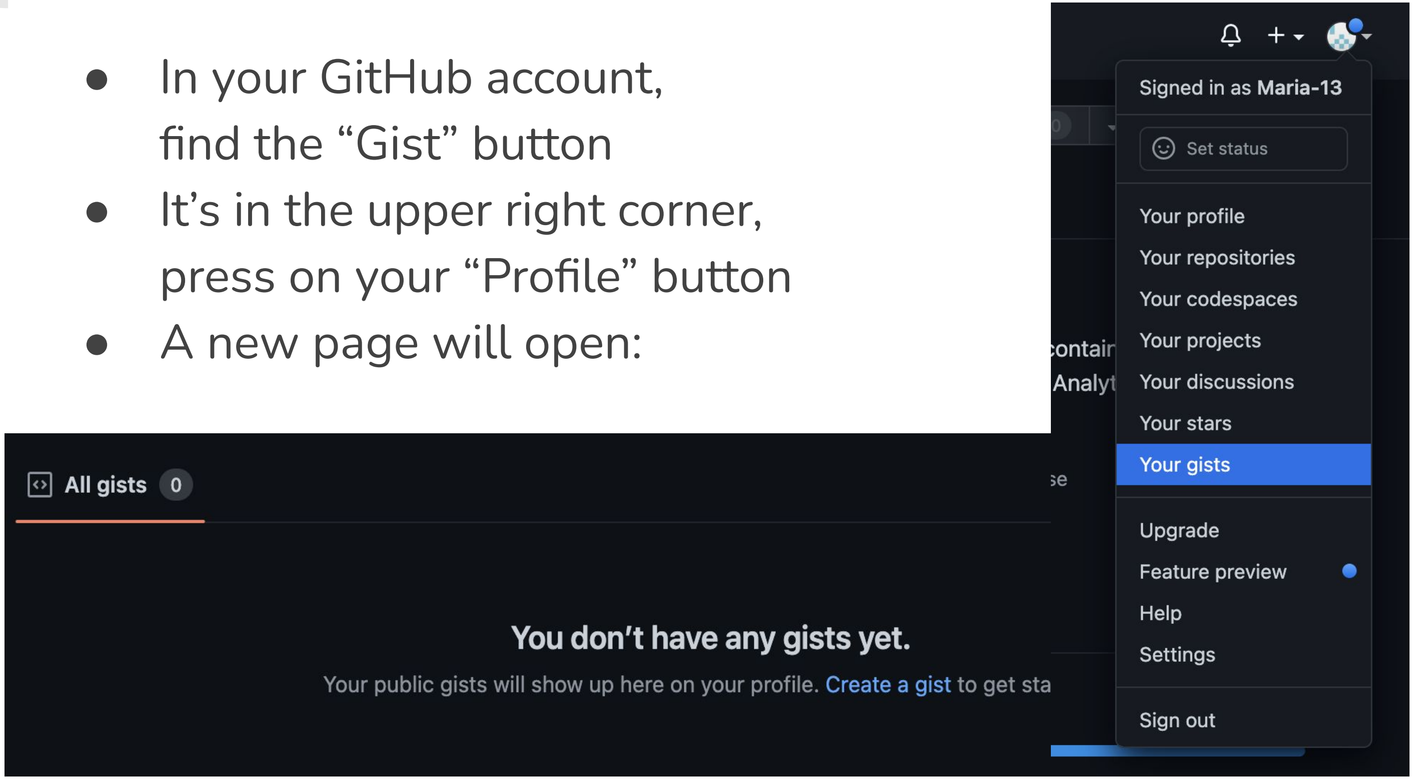
! Unless you reached to me through GitHub Discussions in advanced and told me that you want to use a specific number of late days

Course information: homework

- To complete the homework, you need to have your own GitHub account
- Start “watching” the repository: https://github.com/Maria-13/DataAnalytics_R
- In ‘Assignments” folder, go to the Rscript of ‘hw1’
- Copy/paste the code in your RStudio session
- Complete the assignment by adding your comments and script

Course information: homework

- In your GitHub account, find the “Gist” button
- It’s in the upper right corner, press on your “Profile” button
- A new page will open:

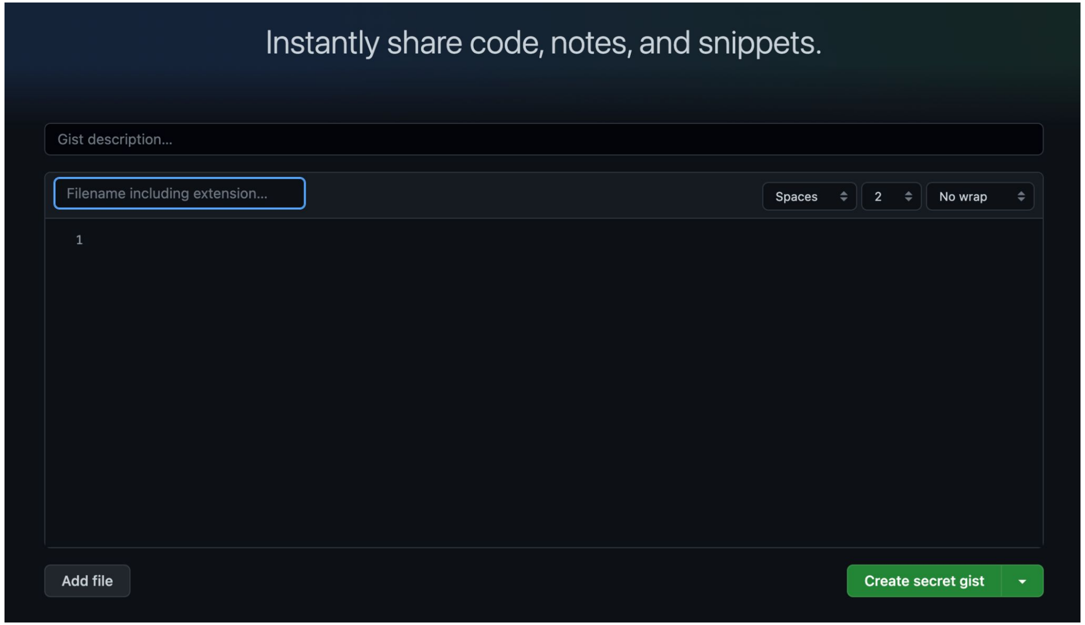


You don’t have any gists yet.

Your public gists will show up here on your profile. Create a gist to get sta

Course information: homework

- Press on “Create a new gist” button


Instantly share code, notes, and snippets.

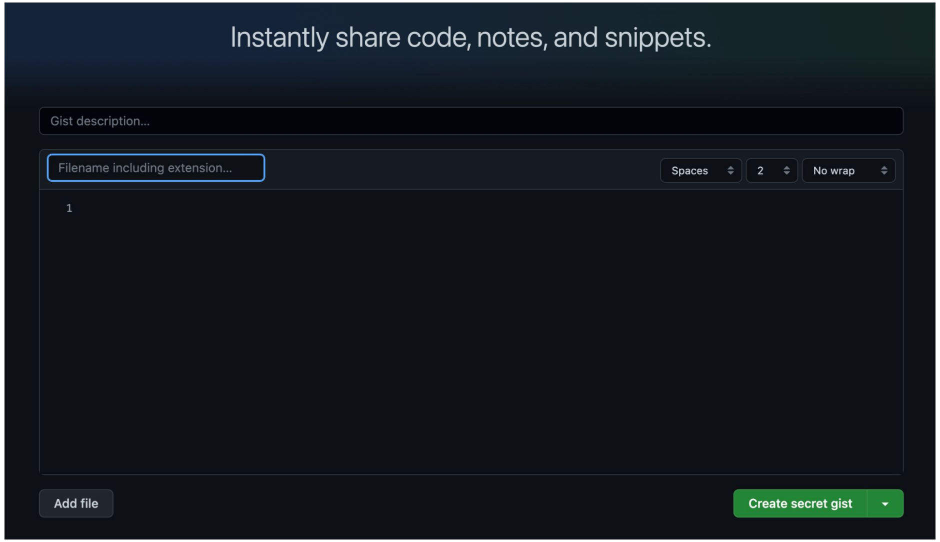
Gist description...

Filename including extension...

Spaces 2 No wrap

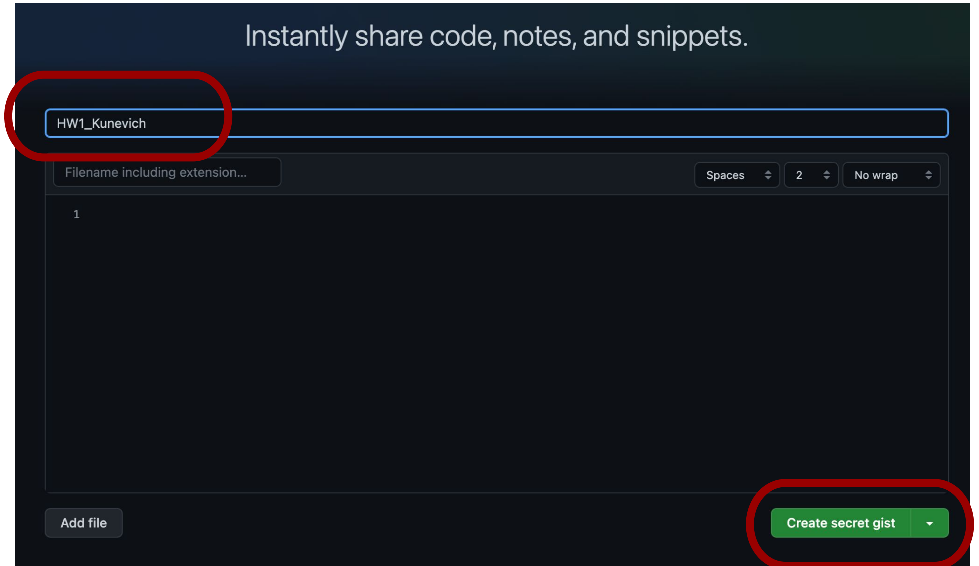
Course information: homework

- You can copy and paste your R code here, or
- Press “Add file” and add your HW1 assignment as an R script file


Instantly share code, notes, and snippets.

Course information: homework

- In Gist description add “HW1_yourlastname”
- In “File name” add “HW1_yourlastname”
- Then press “Create Secret gist”

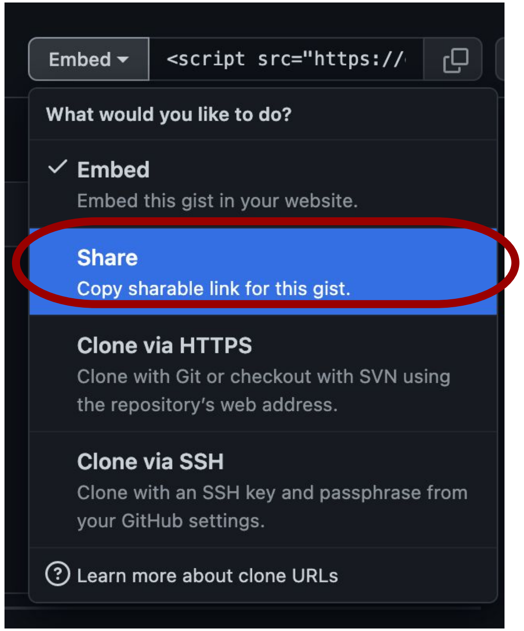

Instantly share code, notes, and snippets.

HW1_Kunevich

Filename including extension...

Course information: homework

- After the Gist has been created, go to the ‘Embed’ button
- It’s in the upper right corner
- Press the button and choose “Share”
- The created link has to be submitted to the “Discussions” panel, as a comment to HW1 thread


Embed
<script src="https://
What would you like to do?
Embed
Embed this gist in your website.
Share
Copy sharable link for this gist.
Clone via HTTPS
Clone with Git or checkout with SVN using the repository’s web address.
Clone via SSH
Clone with an SSH key and passphrase from your GitHub settings.
Learn more about clone URLs

Plan for today

1. Revision of the basic concepts in R
2. Practice session: working with vectors
3. Other data structures in R: factors, lists, arrays, matrices, dataframes
4. “Tidyverse” packages

Recap

What concepts/words do you remember from our first session?

Recap

What concepts/words do you remember from our first session?

- R functions
- Assign operator
- Function arguments
- R object
- Classes of objects
- R data types
- Object name (CASE sensitive)
- Working directory

Recap - Data types in R

Some possible confusion:

Data in R

Data types in R

Variables in R

R objects

Data classes

Data structures in R

Recap

Some confusion about R objects:

- In R, everything is an object!
- a **class** is the blueprint for an object
- There are different classes/types of R objects/variables
- In R, variables are assigned to objects rather than data types

Data types in R:

character, integer, float, long, double

5 basic types of objects / data structures

atomic vectors (+factors), arrays, lists, matrices, data frames

Data structures in R: vectors

- Vector is the simplest data structure in R
- A **numeric** vector is a single entity consisting of a collection of numbers (any real number)
- An **integer** vector is an atomic vector which consists of integers or NA items
- Numbers: **integer** (whole numbers), **double** (double precision for numeric values)
- When you store a number as a variable in R, it converts the number into a 'double' value, e.g 5.00
- Let's explore '*as.double*' and '*is.double*' functions in R

Data structures in R: vectors

- Vector is the simplest data structure in R
- A **logical** vector can have the values TRUE, FALSE, and NA (for “not available”): Abbreviations: **T, F**
- Logical vectors are generated by **conditions**
- For example,   *temp <- x > 13*

sets *temp* as a vector of the same length as *x* with values FALSE corresponding to elements of *x* where the condition is not met and TRUE where it is met

- The logical operators are <, <=, >, >=, == for exact equality and *!=* for inequality

Data structures in R: vectors

- Vector is the simplest data structure in R
- A **character** vector is a string of characters or individual characters, quoted in “” or ‘’
- Single and double quotes can be used **interchangeably** in R
- For example, “name1”, “name2” or ‘name1’, ‘name2’
- However, double quotes are **preferred** (and character variables are printed using double quotes)
- Explore 'as.character' and 'is.character' functions in R

Data structures in R: vectors - factors

- Vector is the simplest data structure in R
- A **factor** vector includes categorical variables or qualitative variables
- A **factor** is a **special character vector** where the elements have pre-defined groups or ‘levels’
- By default, **factors go in alphanumerical order**
- Don’t use *as.factor* function, use *factor()*, even when re-creating a factor
- Use the *levels()* function to check the levels if you need

Data structures in R: class() and typeof()

- Let’s look at **class()** and **typeof()** functions -> a class is the blueprint for an object
- the **class()** is used to define/identify what "type" an object is from the point of view of object-oriented programming in R
- **typeof()** gives the "type" of object from R's point of view
- The main difference between *class* and *typeof* is that the first can be **defined by the user**, but the type cannot

Data structures in R: lists

- ***List*** is another type of object in R programming
- Lists are R objects that contain heterogeneous data types such as strings, numbers, vectors or another list
- The list is created using the list() function
- Let’s create and explore lists in R
- You can access the list items by referring to its **index number**, inside brackets. The first item has index 1, the second item has index 2, and so on

Data structures in R: arrays

- **Arrays** can be considered as a multiply subscripted collection of data entries, for example numeric
- *array()* function is used to create **n-dimensional** array. This function takes *dim* attribute as an argument and creates required length of each dimension as specified in the attribute
- A dimension vector is a vector of non-negative integers
- You can access the array elements also by referring to the **index** position
- Let's explore array() function in R

Data structures in R: matrices and data frames

- **Matrices** are used in R to store values as 2-Dimensional arrays
- data, number of rows and columns are defined in the matrix() function
- **Data frames** are *2-dimensional* tabular data object in R
- They consists of multiple columns and each column represents a vector
- Columns in data frames can have different modes of data unlike matrices

Data structures in R: recap
‘Tidyverse’ packages

- Vectors (factors)
- Lists
- Arrays
- Matrices
- Data frames

Very important packages for data analysis:

https://www.tidyverse.org/packages/

Download and explore for our future sessions!

Data structures in R: matrices and data frames

- **In Session 4** we will discuss **Matrices** in more detail
- Then revise some basic **concepts for statistical analysis**
- **In Session 5** we will discuss **Data frames** in more detail
- And we will explore different packages for importing and visualising data in R

Thank you!