THE UNIVERSITY OF TEXAS AT AUSTIN
McCOMBS SCHOOL OF BUSINESS

Introduction to R

Lecture 2

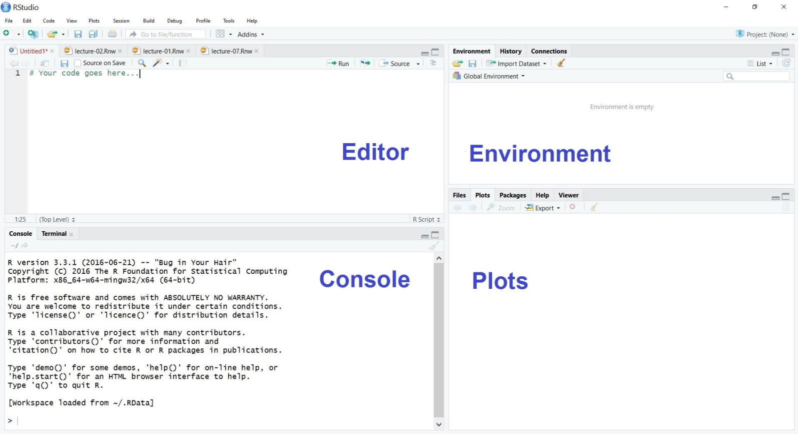
STA 371G

RStudio

R is the language, which we access through RStudio (interface).

RStudio

R is the language, which we access through RStudio (interface).



```
# Your code goes here...
```

```
R version 3.3.1 (2016-06-21) -- "Bug in Your Hair"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

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.

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.

[Workspace loaded from ~/.RData]

>
```

RStudio

- Console: This is where calculations/code are passed to R and results are observed.

RStudio

- Console: This is where calculations/code are passed to R and results are observed.
- Editor: It is not practical to write long calculations/code in console. We write them in the editor, and "Run" to pass to the console.

RStudio

- Console: This is where calculations/code are passed to R and results are observed.
- Editor: It is not practical to write long calculations/code in console. We write them in the editor, and "Run" to pass to the console.
- Environment: All data sets/variables we define can be found here.

RStudio

- Console: This is where calculations/code are passed to R and results are observed.
- Editor: It is not practical to write long calculations/code in console. We write them in the editor, and "Run" to pass to the console.
- Environment: All data sets/variables we define can be found here.
- Plots: When we plot things, they will first appear here.

Let’s get started...

Suppose you want to calculate your course grade.

Assignment	Weight	Grade
Class participation	5%	91
Reading assignments	5%	95
Homework	15%	86
Project	15%	83
Midterm 1	20%	88
Midterm 2	20%	84
Final exam	20%	76

Using the console

Try this in the console (enter all on one line and press Enter):

```
0.05*91 + 0.05*95 + 0.15*86 + 0.15*83 +
  0.2*88 + 0.2*84 + 0.2*76

[1] 84.25
```

Using the console

We can also assign the result of a calculation to a variable. Here we’ll create a variable called `grade` which will contain the result of our calculation:

```r
grade <- 0.05*91 + 0.05*95 + 0.15*86 + 0.15*83 +
  0.2*88 + 0.2*84 + 0.2*76
```

Using the console

We can also assign the result of a calculation to a variable. Here we'll create a variable called `grade` which will contain the result of our calculation:

```r
grade <- 0.05*91 + 0.05*95 + 0.15*86 + 0.15*83 +
  0.2*88 + 0.2*84 + 0.2*76
```

Now we can do calculations using that variable. For example, what will our grade be if the grades are curved up by 5 points?

```r
grade + 10

[1] 94.25
```

Using the editor

In R, an *vector* is just a list of numbers. Let’s redo the calculation using vectors:

```r
# This is the same calculation, using vectors.
weights <- c(0.05, 0.05, 0.15, 0.15, 0.2, 0.2, 0.2)
grades <- c(91, 95, 86, 83, 88, 84, 76)
weighted.grades <- weights * grades
my371 <- sum(weighted.grades)
```

The multiplication is “element-wise,” meaning that the corresponding elements in each vector are multiplied.

Using the editor

In R, an *vector* is just a list of numbers. Let's redo the calculation using vectors:

```r
# This is the same calculation, using vectors.
weights <- c(0.05, 0.05, 0.15, 0.15, 0.2, 0.2, 0.2)
grades <- c(91, 95, 86, 83, 88, 84, 76)
weighted.grades <- weights * grades
my371 <- sum(weighted.grades)
```

The multiplication is "element-wise," meaning that the corresponding elements in each vector are multiplied. Then, the `sum` function adds up all the elements in the resulting vector.

Working with tabular data

Many data sets we will work with are in tabular format, saved in “CSV” files (CSV = comma-separated values).

Working with tabular data

Many data sets we will work with are in tabular format, saved in “CSV” files (CSV = comma-separated values).

Let’s analyze the passenger data from the Titanic disaster. Load the file by copying and pasting the command from Learning Catalytics, and then type `View(titanic)` to view the data set.

Working with tabular data

Many data sets we will work with are in tabular format, saved in "CSV" files (CSV = comma-separated values).

Let's analyze the passenger data from the Titanic disaster. Load the file by copying and pasting the command from Learning Catalytics, and then type `View(titanic)` to view the data set.

This data has five variables:

- Name: The name of the passenger
- PClass: The class of the passenger (1st, 2nd, etc)
- Age: The age of the passenger, in years
- Sex: The sex of the passenger
- Survived: Whether the passenger survived the disaster

Working with tabular data

`$` is used to refer to a particular column in the data, such as `titanic$Name`.

Working with tabular data

`$` is used to refer to a particular column in the data, such as `titanic$Name`.

To access to an element in a particular position, e.g., row 1, column 4, use `titanic[1,4]`.

Exploring Categorical Variables

The dataset has both quantitative and categorical data. (What’s the difference?)

Exploring Categorical Variables

The dataset has both quantitative and categorical data. (What’s the difference?)

Let’s explore the categorical variables through some frequency tables.
Let’s say we want to get a frequency table of the number of passengers by class:

```
table(titanic$PClass)



1st 2nd 3rd
323 279 711
```

Exploring Categorical Variables

What is more interesting is a two-way table showing how many people survived in each passenger class. We'll assign the table to a variable for later use!

```
class_survival <- table(titanic$Survived, titanic$PClass)
class_survival
```

```
      1st 2nd 3rd
  No  130 160 573
  Yes 193 119 138
```

Exploring Categorical Variables

To get a better sense of the data, let’s calculate the survival percentage for each passenger class.

```
prop.table(class_survival, 2)
```

```
          1st       2nd       3rd
No  0.4024768 0.5734767 0.8059072
Yes 0.5975232 0.4265233 0.1940928
```

Exploring Categorical Variables

To get a better sense of the data, let’s calculate the survival percentage for each passenger class.

```
prop.table(class_survival, 2)


          1st       2nd       3rd
No  0.4024768 0.5734767 0.8059072
Yes 0.5975232 0.4265233 0.1940928
```

It looks like one’s chance of survival highly depended on his/her passenger class!

Slicing the data

One very common operation is slicing the data, i.e., selecting the portion that satisfy certain conditions.

Slicing the data

One very common operation is slicing the data, i.e., selecting the portion that satisfy certain conditions.

For example, we can select the rows that belong to female passenger data.

```r
female.passengers <- subset(titanic, Sex == "female")
```

Slicing the data

One very common operation is slicing the data, i.e., selecting the portion that satisfy certain conditions.

For example, we can select the rows that belong to female passenger data.

```r
female.passengers <- subset(titanic, Sex == "female")
```

This means: in the titanic dataset, select rows where Sex is `female` and save the resulting table to the `female.passengers` variable.

Slicing the data

We can create more complex conditions – what do you think this does?

```
my.data <- subset(titanic, Sex == "female" &
                           PClass == "1st")
```

Slicing the data

We can create more complex conditions – what do you think this does?

```
my.data <- subset(titanic, Sex == "female" &
                           PClass == "1st")
```

We can use the `nrow` function to count the number of rows in a dataset:

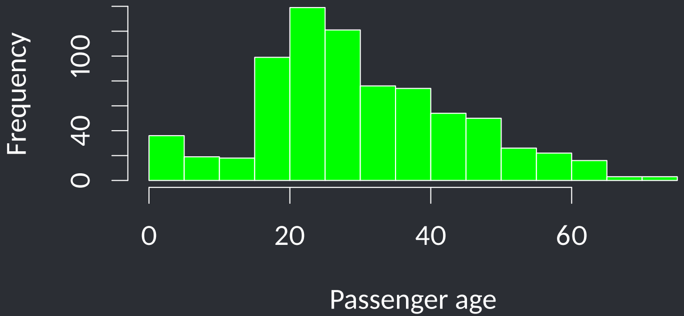
```
nrow(my.data)

[1] 143
```

There were 143 women in first class on the Titanic!

Exploring quantitative data

Let’s look into age distribution of the passengers.

```
hist(titanic$Age, col="green",
     xlab="Passenger age", main="")
```



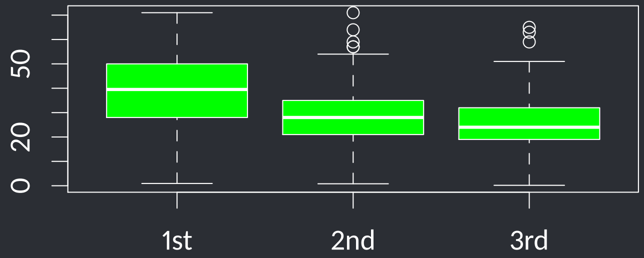
Exploring quantitative data

We can look at the relationship between a quantitative variable and a categorical one by generating side-by-side boxplots to compare the distribution of the quantiative variable for each value of the categorical variable:

Exploring quantitative data

We can look at the relationship between a quantitative variable and a categorical one by generating side-by-side boxplots to compare the distribution of the quantiative variable for each value of the categorical variable:

```
boxplot(Age ~ PClass, data=titanic, col="green", main="")
```


1st
2nd
3rd