# Eric Pitman Summer Workshop in Computational Science

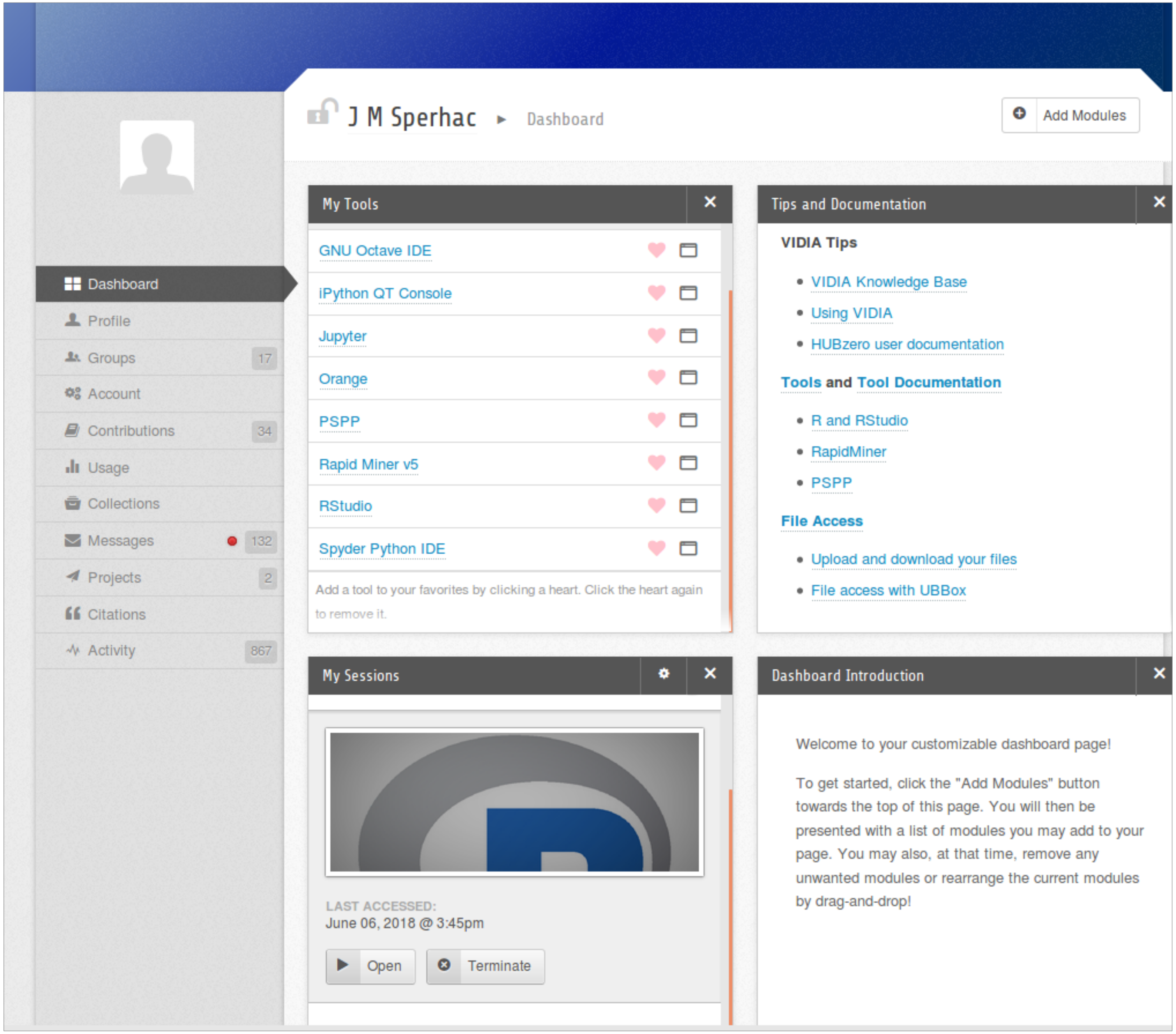
## Intro to R

## RStudio Tips

CENTER FOR COMPUTATIONAL RESEARCH

University at Buffalo
The State University of New York

# VIDIA Dashboard



J M Sperhac ▸ Dashboard

Add Modules

- Dashboard
- Profile
- Groups 17
- Account
- Contributions 34
- Usage
- Collections
- Messages 132
- Projects 2
- Citations
- Activity 867

**My Tools**

- GNU Octave IDE
- iPython QT Console
- Jupyter
- Orange
- PSPP
- Rapid Miner v5
- RStudio
- Spyder Python IDE

Add a tool to your favorites by clicking a heart. Click the heart again to remove it.

**Tips and Documentation**

**VIDIA Tips**

- VIDIA Knowledge Base
- Using VIDIA
- HUBzero user documentation

**Tools and Tool Documentation**

- R and RStudio
- RapidMiner
- PSPP

**File Access**

- Upload and download your files
- File access with UBBox

**My Sessions**

LAST ACCESSED:
June 06, 2018 @ 3:45pm

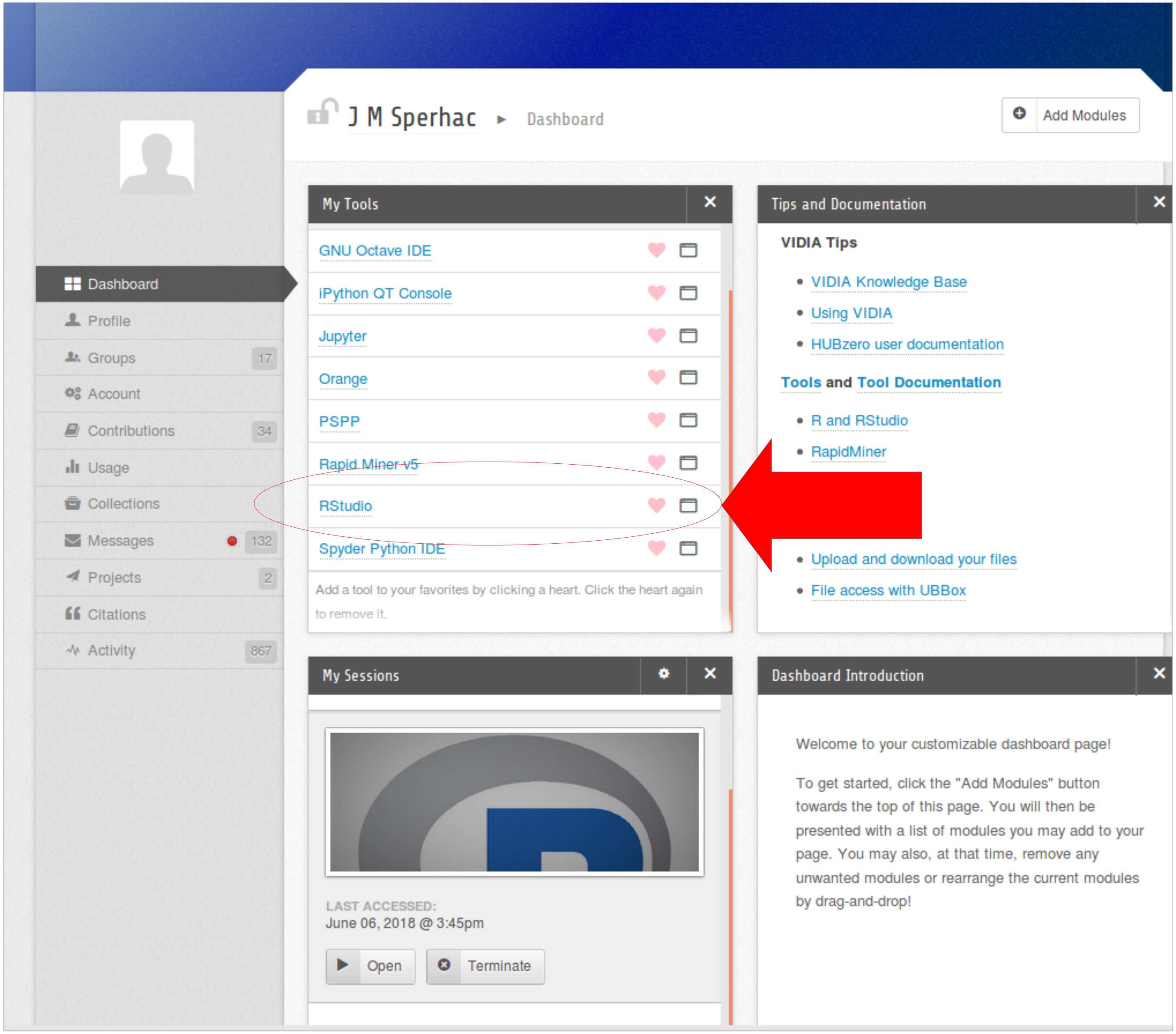
▶ Open ⊗ Terminate

**Dashboard Introduction**

Welcome to your customizable dashboard page!

To get started, click the "Add Modules" button towards the top of this page. You will then be presented with a list of modules you may add to your page. You may also, at that time, remove any unwanted modules or rearrange the current modules by drag-and-drop!

# VIDIA Dashboard: RStudio Tool

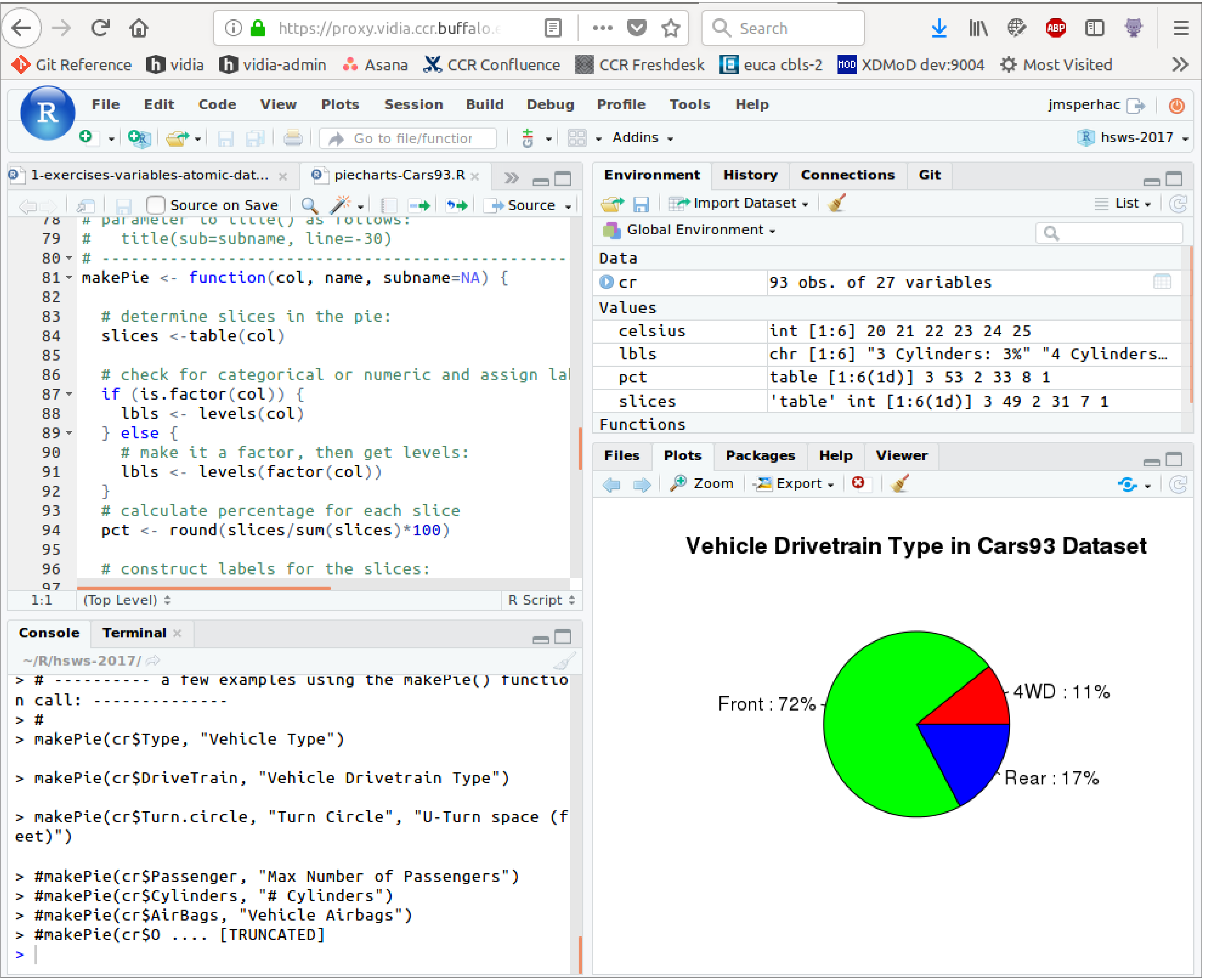


Add a tool to your favorites by clicking a heart. Click the heart again to remove it.

Welcome to your customizable dashboard page!

To get started, click the "Add Modules" button towards the top of this page. You will then be presented with a list of modules you may add to your page. You may also, at that time, remove any unwanted modules or rearrange the current modules by drag-and-drop!

# RStudio Interactive Development Environment (IDE)



```
# parameter to title() as follows:
#   title(sub=subname, line=-30)
# ----------------------------------------------
makePie <- function(col, name, subname=NA) {

  # determine slices in the pie:
  slices <-table(col)

  # check for categorical or numeric and assign lal
  if (is.factor(col)) {
    lbls <- levels(col)
  } else {
    # make it a factor, then get levels:
    lbls <- levels(factor(col))
  }
  # calculate percentage for each slice
  pct <- round(slices/sum(slices)*100)

  # construct labels for the slices:
```

| | |
|---|---|
| Data | |
| cr | 93 obs. of 27 variables |
| Values | |
| celsius | int [1:6] 20 21 22 23 24 25 |
| lbls | chr [1:6] "3 Cylinders: 3%" "4 Cylinders… |
| pct | table [1:6(1d)] 3 53 2 33 8 1 |
| slices | 'table' int [1:6(1d)] 3 49 2 31 7 1 |
| Functions | |

```
> # ---------- a few examples using the makePie() functio
n call: -------------
> #
> makePie(cr$Type, "Vehicle Type")

> makePie(cr$DriveTrain, "Vehicle Drivetrain Type")

> makePie(cr$Turn.circle, "Turn Circle", "U-Turn space (f
eet)")

> #makePie(cr$Passenger, "Max Number of Passengers")
> #makePie(cr$Cylinders, "# Cylinders")
> #makePie(cr$AirBags, "Vehicle Airbags")
> #makePie(cr$O .... [TRUNCATED]
>
```

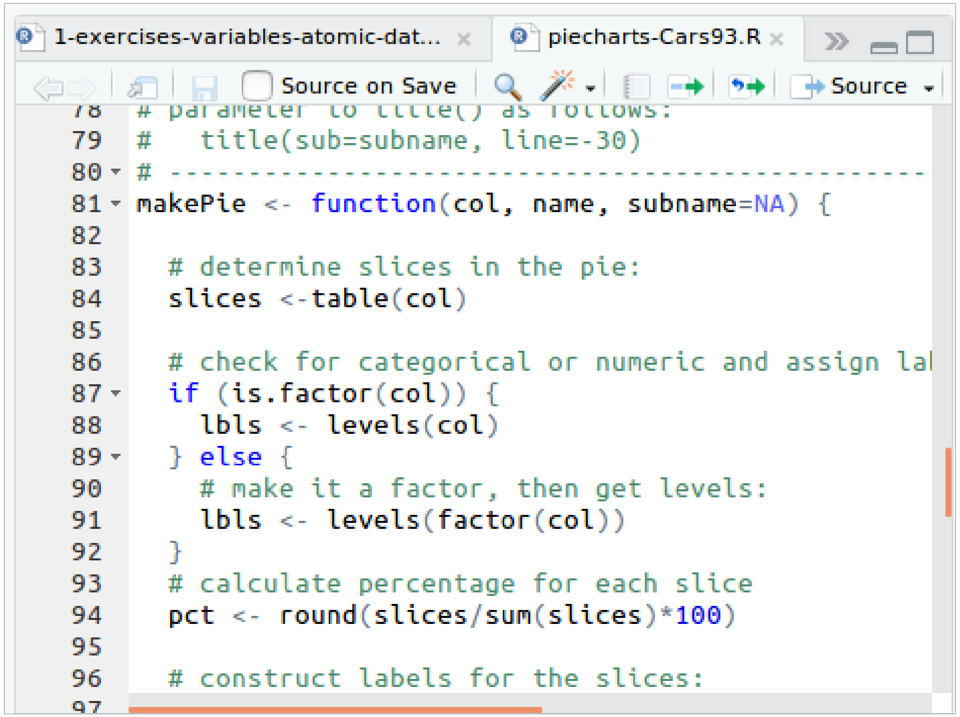
1. Editor

2. Workspace (Variables) and History

3. Plots, etc.

4. Console

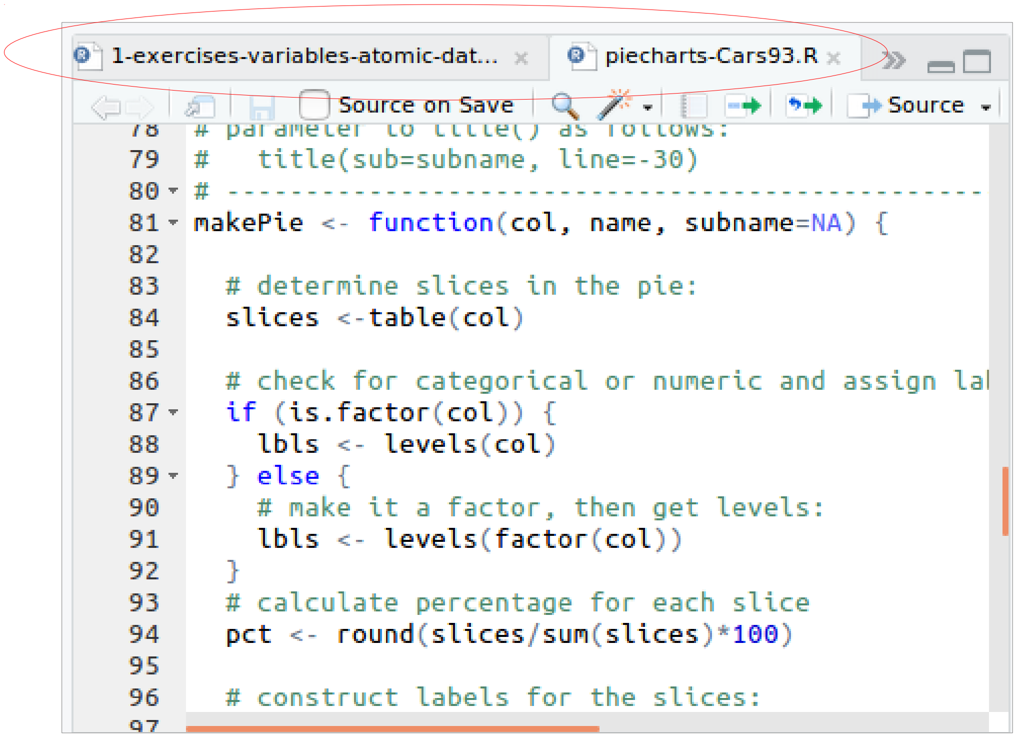
# Editor window



```
# parameter to title() as follows:
#   title(sub=subname, line=-30)
# -----------------------------------------------
makePie <- function(col, name, subname=NA) {

  # determine slices in the pie:
  slices <-table(col)

  # check for categorical or numeric and assign lal
  if (is.factor(col)) {
    lbls <- levels(col)
  } else {
    # make it a factor, then get levels:
    lbls <- levels(factor(col))
  }
  # calculate percentage for each slice
  pct <- round(slices/sum(slices)*100)

  # construct labels for the slices:
```

Select, view, edit, save, and execute scripts.

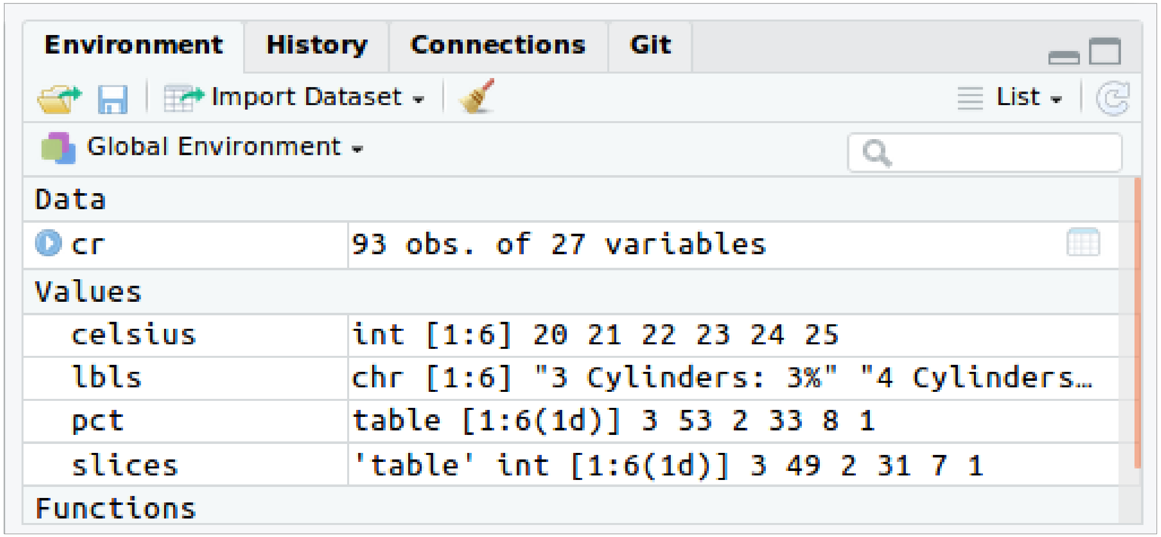
# Editor window



```
# parameter to title() as follows:
#   title(sub=subname, line=-30)
# -----------------------------------------------
makePie <- function(col, name, subname=NA) {

  # determine slices in the pie:
  slices <-table(col)

  # check for categorical or numeric and assign lal
  if (is.factor(col)) {
    lbls <- levels(col)
  } else {
    # make it a factor, then get levels:
    lbls <- levels(factor(col))
  }
  # calculate percentage for each slice
  pct <- round(slices/sum(slices)*100)

  # construct labels for the slices:
```

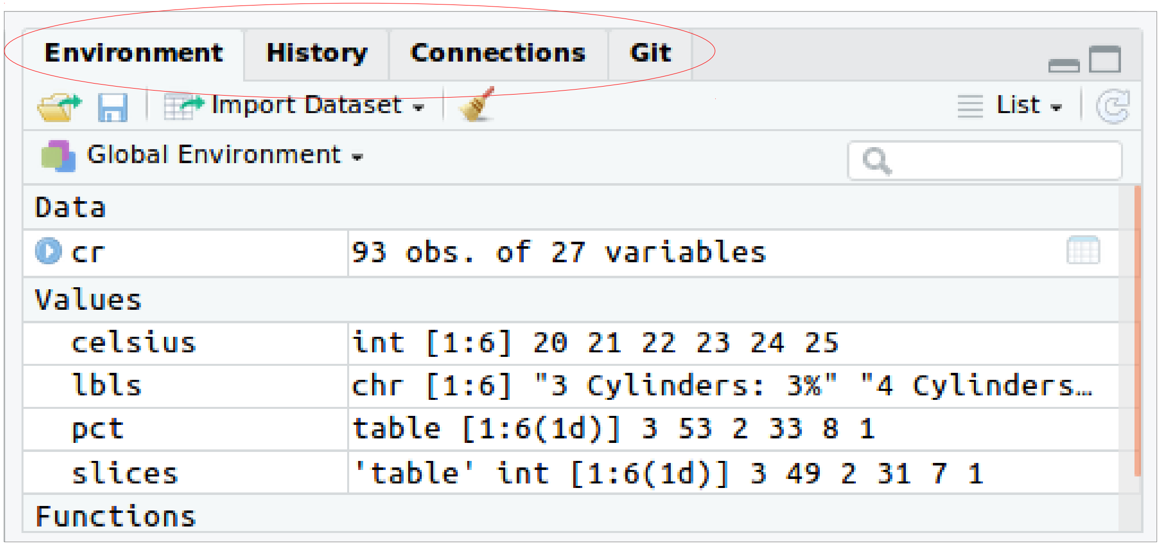
Select, view, edit, save, and execute scripts.

# Workspace and History window



```
Data
cr            93 obs. of 27 variables
Values
  celsius     int [1:6] 20 21 22 23 24 25
  lbls        chr [1:6] "3 Cylinders: 3%" "4 Cylinders…
  pct         table [1:6(1d)] 3 53 2 33 8 1
  slices      'table' int [1:6(1d)] 3 49 2 31 7 1
Functions
```

# Workspace and History window

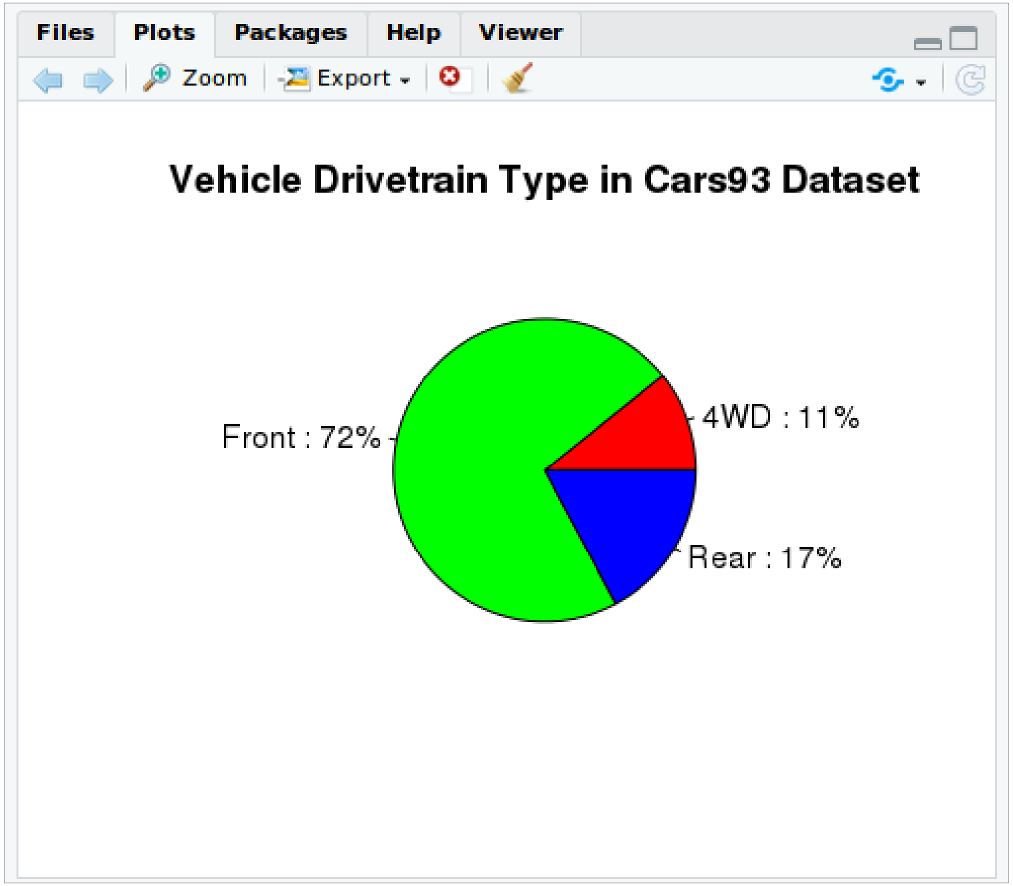


```
Data
cr          93 obs. of 27 variables
Values
  celsius   int [1:6] 20 21 22 23 24 25
  lbls      chr [1:6] "3 Cylinders: 3%" "4 Cylinders…
  pct       table [1:6(1d)] 3 53 2 33 8 1
  slices    'table' int [1:6(1d)] 3 49 2 31 7 1
Functions
```

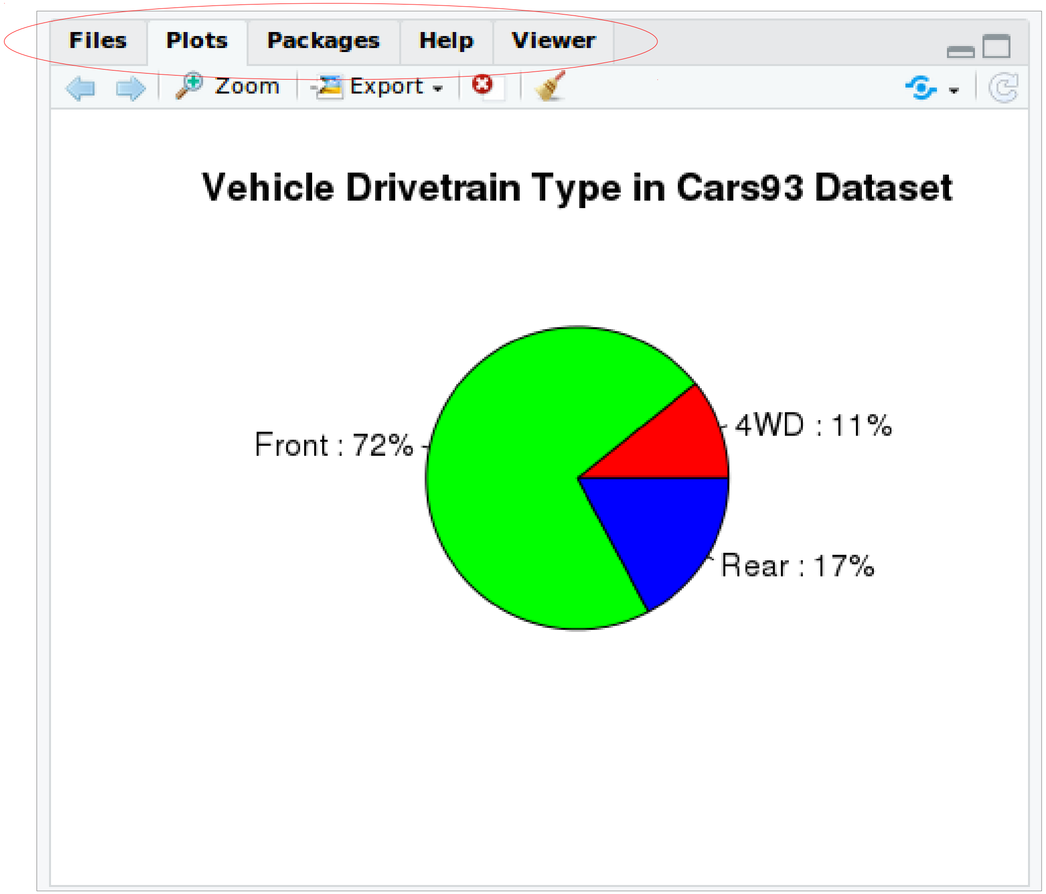
Pick a tab to access:

- Current variables (**Environment**)
- Command **History**
- (Database) **Connections**
- Version control (**Git**) status and commands

# Plot (etc.) window



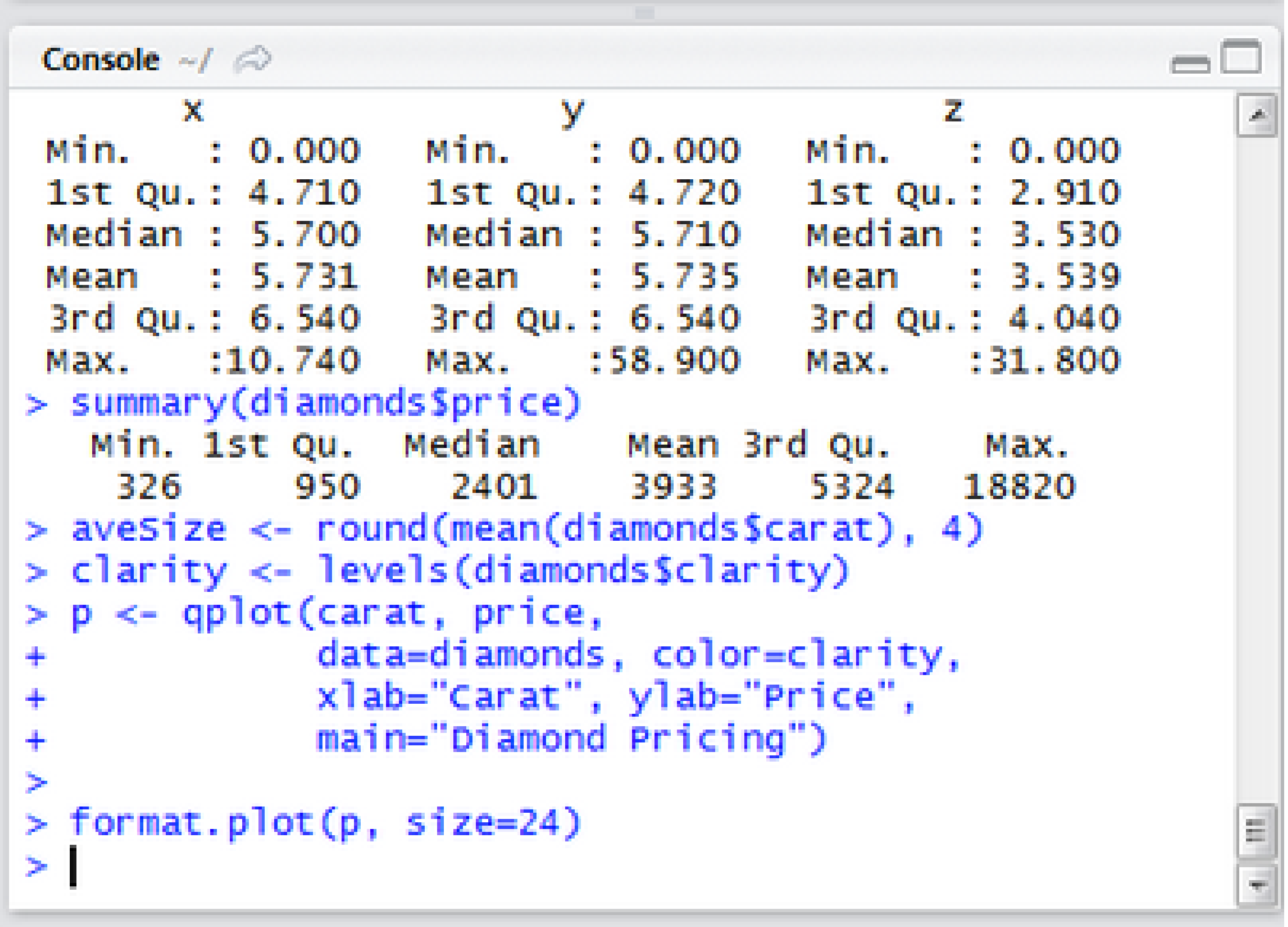
# Plot (etc.) window



Pick a tab to:

- Access **Files** and Directories
- View current **Plots**
- Review loaded and available **Packages**
- Read **Help** and Documentation
- **View** markdown

# Console window

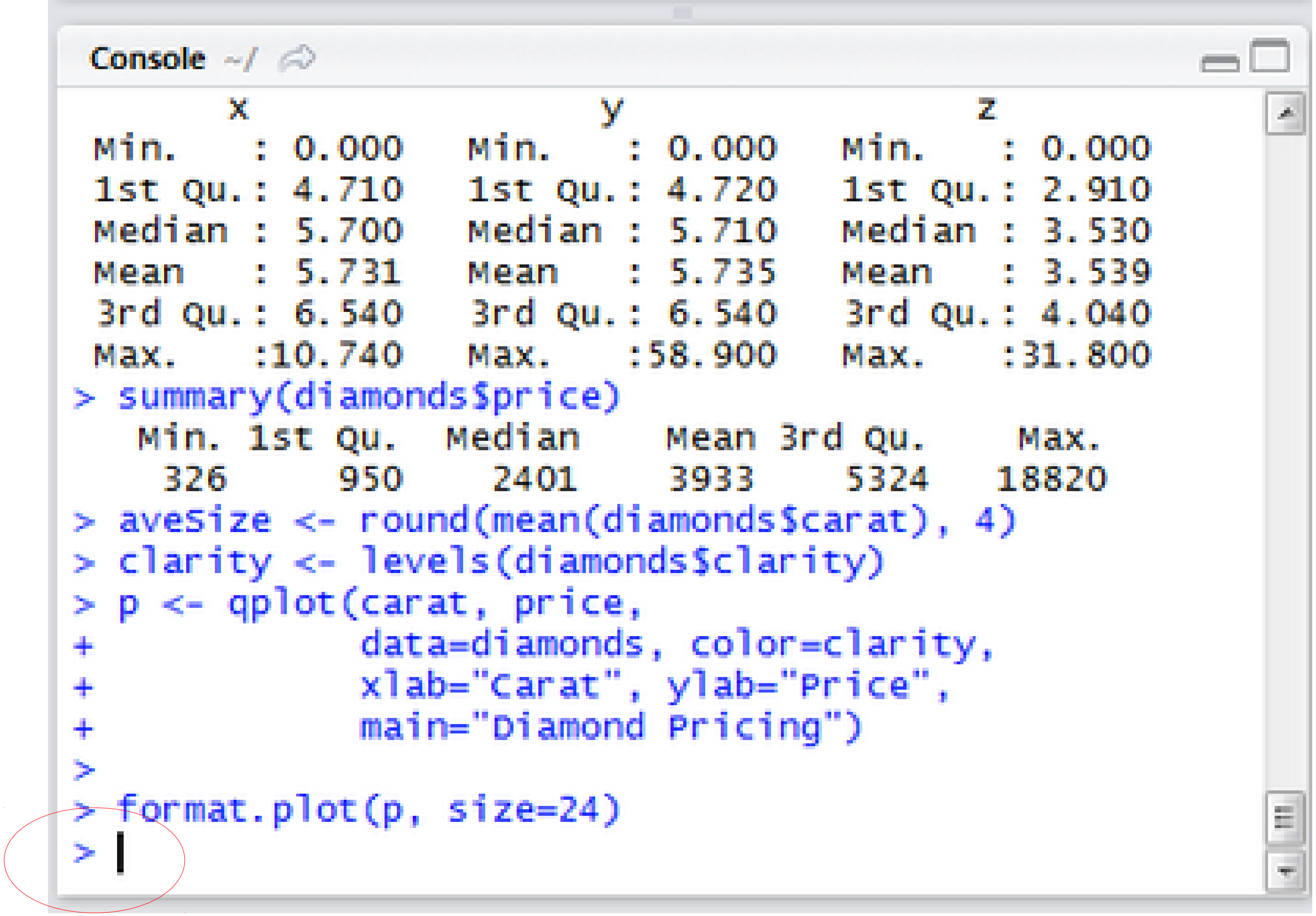


```
       x                 y                 z
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
 1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
 Median : 5.700   Median : 5.710   Median : 3.530
 Mean   : 5.731   Mean   : 5.735   Mean   : 3.539
 3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040
 Max.   :10.740   Max.   :58.900   Max.   :31.800
> summary(diamonds$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    326     950    2401    3933    5324   18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+            data=diamonds, color=clarity,
+            xlab="Carat", ylab="Price",
+            main="Diamond Pricing")
>
> format.plot(p, size=24)
> 
```

On the command line:

- Issue commands
- Review the results
- Puzzle over error messages

# Console window



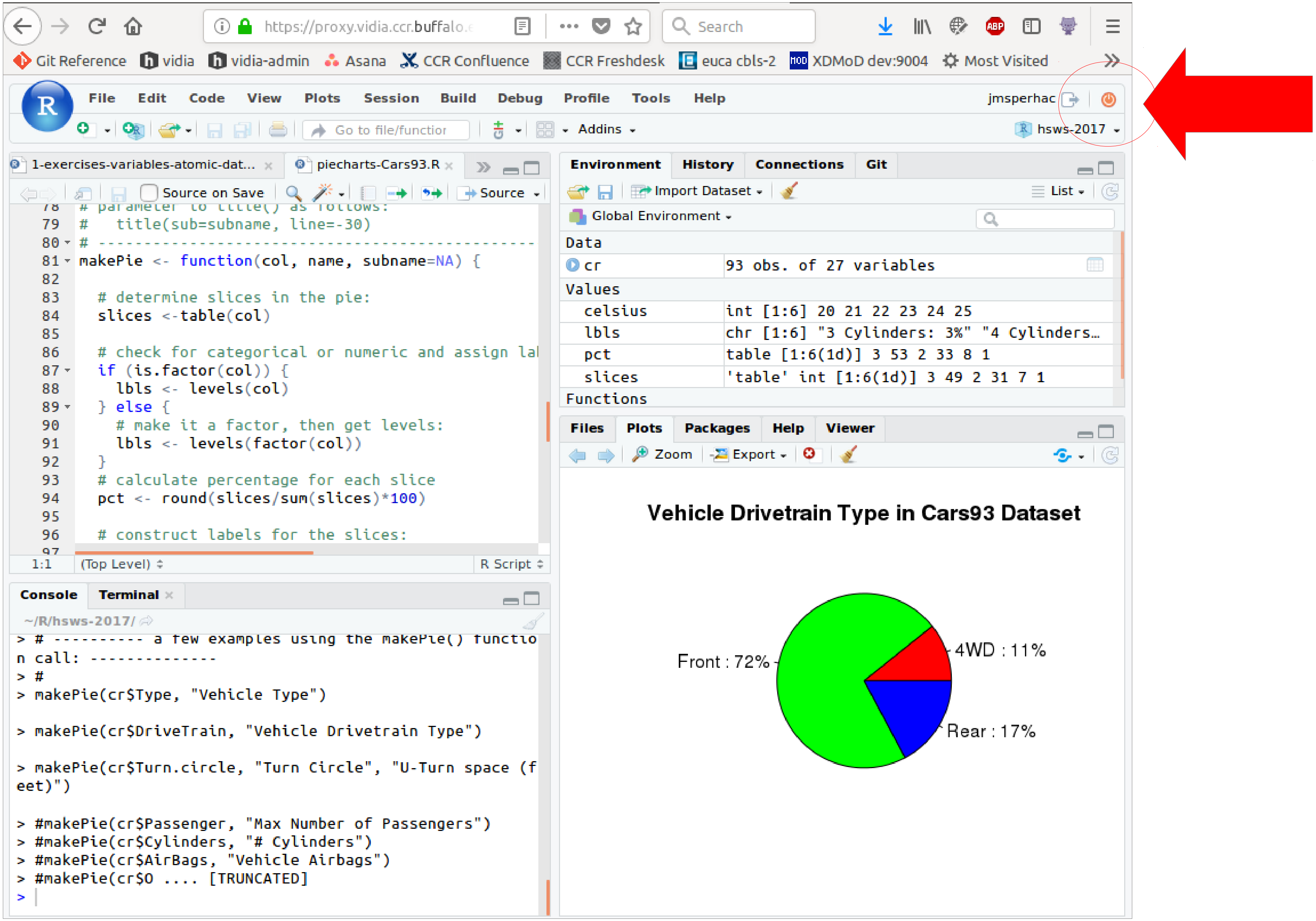
```
        x                 y                 z        
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000  
 1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910  
 Median : 5.700   Median : 5.710   Median : 3.530  
 Mean   : 5.731   Mean   : 5.735   Mean   : 3.539  
 3rd Qu.: 6.540   3rd Qu.: 6.540   3rd Qu.: 4.040  
 Max.   :10.740   Max.   :58.900   Max.   :31.800  
> summary(diamonds$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    326     950    2401    3933    5324   18820 
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+            data=diamonds, color=clarity,
+            xlab="Carat", ylab="Price",
+            main="Diamond Pricing")
> 
> format.plot(p, size=24)
> |
```

Command prompt

On the command line:

- Issue commands
- Review the results
- Puzzle over error messages

# Quitting RStudio

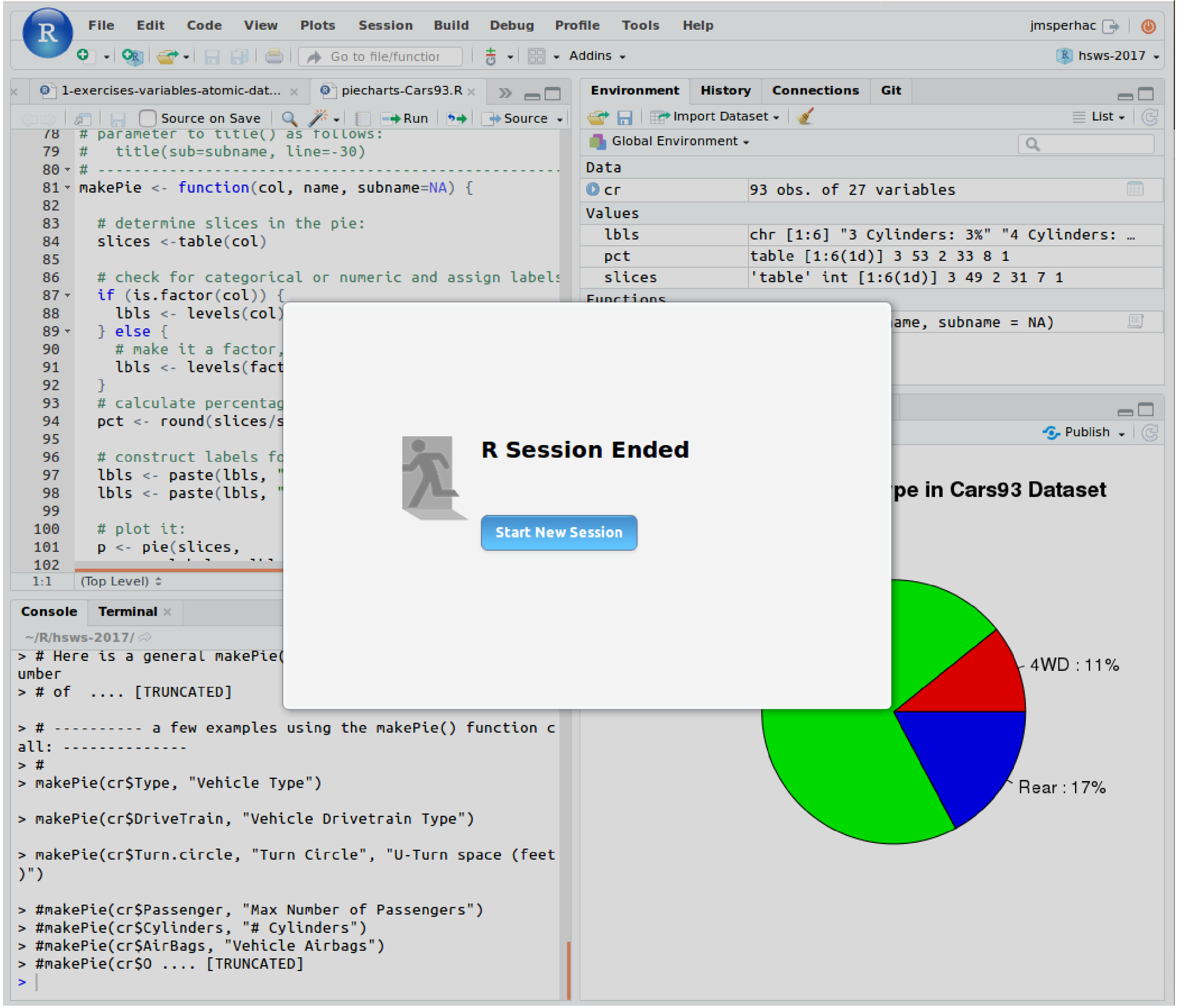


```
#   parameter to title() as follows:
#   title(sub=subname, line=-30)
# --------------------------------------------------
makePie <- function(col, name, subname=NA) {

  # determine slices in the pie:
  slices <-table(col)

  # check for categorical or numeric and assign lal
  if (is.factor(col)) {
    lbls <- levels(col)
  } else {
    # make it a factor, then get levels:
    lbls <- levels(factor(col))
  }
  # calculate percentage for each slice
  pct <- round(slices/sum(slices)*100)

  # construct labels for the slices:
```

```
Data
cr        93 obs. of 27 variables
Values
celsius   int [1:6] 20 21 22 23 24 25
lbls      chr [1:6] "3 Cylinders: 3%" "4 Cylinders…
pct       table [1:6(1d)] 3 53 2 33 8 1
slices    'table' int [1:6(1d)] 3 49 2 31 7 1
Functions
```

```
> # ---------- a few examples using the makePie() functio
n call: -------------
> #
> makePie(cr$Type, "Vehicle Type")

> makePie(cr$DriveTrain, "Vehicle Drivetrain Type")

> makePie(cr$Turn.circle, "Turn Circle", "U-Turn space (f
eet)")

> #makePie(cr$Passenger, "Max Number of Passengers")
> #makePie(cr$Cylinders, "# Cylinders")
> #makePie(cr$AirBags, "Vehicle Airbags")
> #makePie(cr$O .... [TRUNCATED]
> 
```

# Quitting RStudio



```
# parameter to title() as follows:
#   title(sub=subname, line=-30)
# -------------------------------------------------
makePie <- function(col, name, subname=NA) {

  # determine slices in the pie:
  slices <-table(col)

  # check for categorical or numeric and assign labels
  if (is.factor(col)) {
    lbls <- levels(col)
  } else {
    # make it a factor,
    lbls <- levels(fact
  }
  # calculate percentag
  pct <- round(slices/s

  # construct labels fo
  lbls <- paste(lbls, "
  lbls <- paste(lbls, "

  # plot it:
  p <- pie(slices,
```

Data

| | |
|---|---|
| cr | 93 obs. of 27 variables |

Values

| | |
|---|---|
| lbls | chr [1:6] "3 Cylinders: 3%" "4 Cylinders: … |
| pct | table [1:6(1d)] 3 53 2 33 8 1 |
| slices | 'table' int [1:6(1d)] 3 49 2 31 7 1 |

Functions

ame, subname = NA)

```
> # Here is a general makePie(
umber
> # of  .... [TRUNCATED]

> # ---------- a few examples using the makePie() function c
all: --------------
> #
> makePie(cr$Type, "Vehicle Type")

> makePie(cr$DriveTrain, "Vehicle Drivetrain Type")

> makePie(cr$Turn.circle, "Turn Circle", "U-Turn space (feet
)")

> #makePie(cr$Passenger, "Max Number of Passengers")
> #makePie(cr$Cylinders, "# Cylinders")
> #makePie(cr$AirBags, "Vehicle Airbags")
> #makePie(cr$O .... [TRUNCATED]
>
```

**R Session Ended**

Start New Session

# R Practical Matters

- R is case sensitive (R != r)
- Command line prompt is >
- To run R code: use command line, or save script and source(“script_name”)
- To separate commands, use ; or a newline
- The # character marks a non-executed *comment*
- To display help files:

  ?<command-name> or ??<command-name>

# RStudio basics and tips

- Up-arrow and history pane: access and edit previous commands
- You can change window size in the IDE by dragging window borders
- Ctrl-L clears the console window
- Broom icon clears Workspace or Plots
- Is your Project loaded? Check upper right.

# R ...is free

If you want to experiment further with R and RStudio, you can install them on your favorite operating system at home.

First, install R:

http://cran.r-project.org/

Then, install the RStudio IDE:

http://www.rstudio.com/ide/