Advanced Scripting
R - Grundlagen

FH Hagenberg

Gerald Lirk
E-Mail: gerald.lirk@fh-hagenberg.at

Inhalt

- Einleitung
- Datentypen
- Datenhandling
- Programmierung
- plyr, dplyr
- Grafiken (ggplot v.a. 2. Semester)
- statistische Tests
- (Machine Learning)

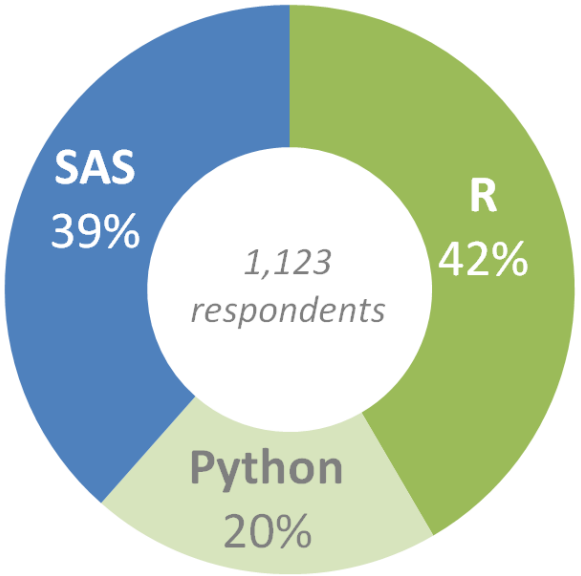
R vs Python
ucanalytics.com

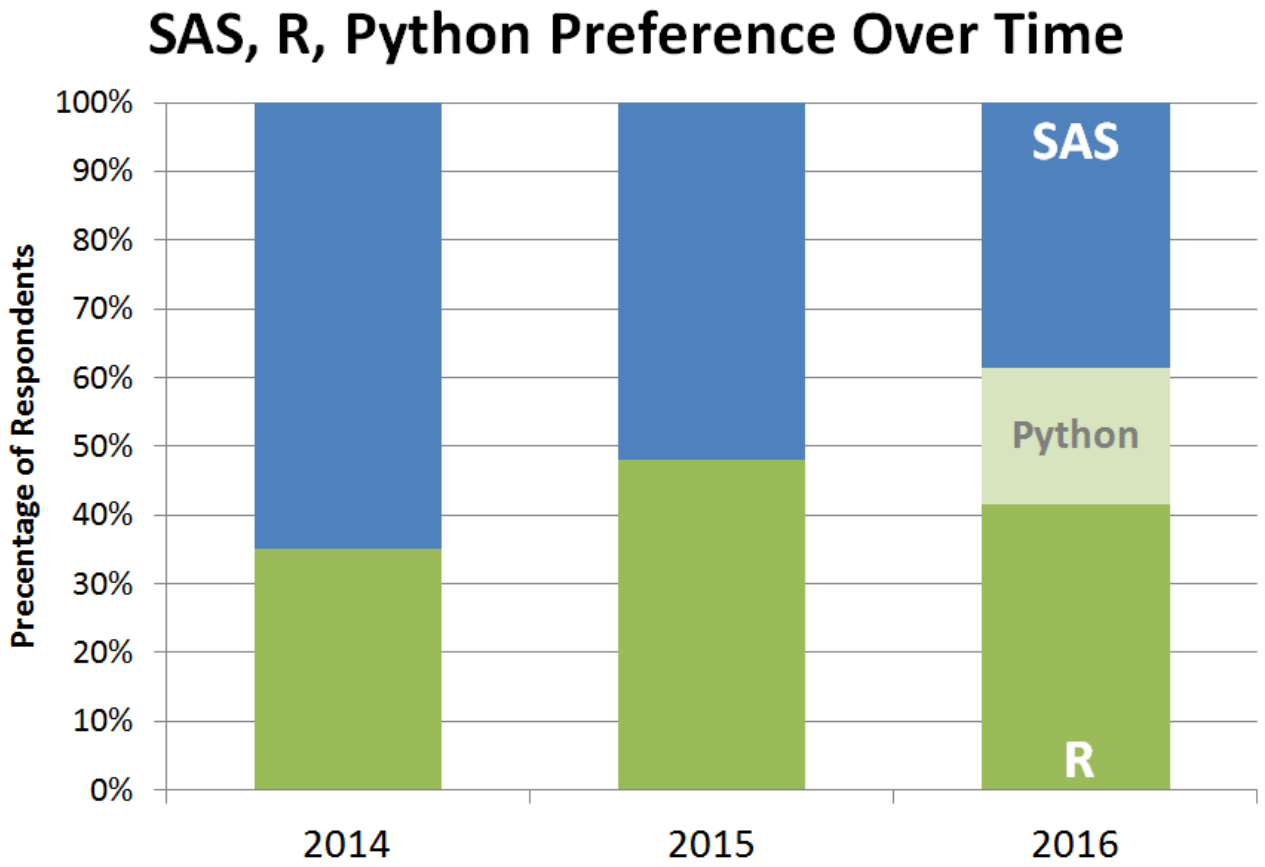
R Qualities	Python Qualities
Use R for analysis, data visualization, and modeling • Offers great flexibility for analysis • R makes it is easy to think while doing your analysis • Constant upgrades and enhancements of analysis packages because of highly active community in statistics and mathematics • Exceptional data visualization tools	Use Python for data preparation, data munging especially for unstructured data like web, images, text etc. • Great flexibility and ability to extract information from free text, websites, and social media sites • Good with mining images and prepare data for analysis • Can handle large volume of data better than R

R vs Python
ucanalytics.com

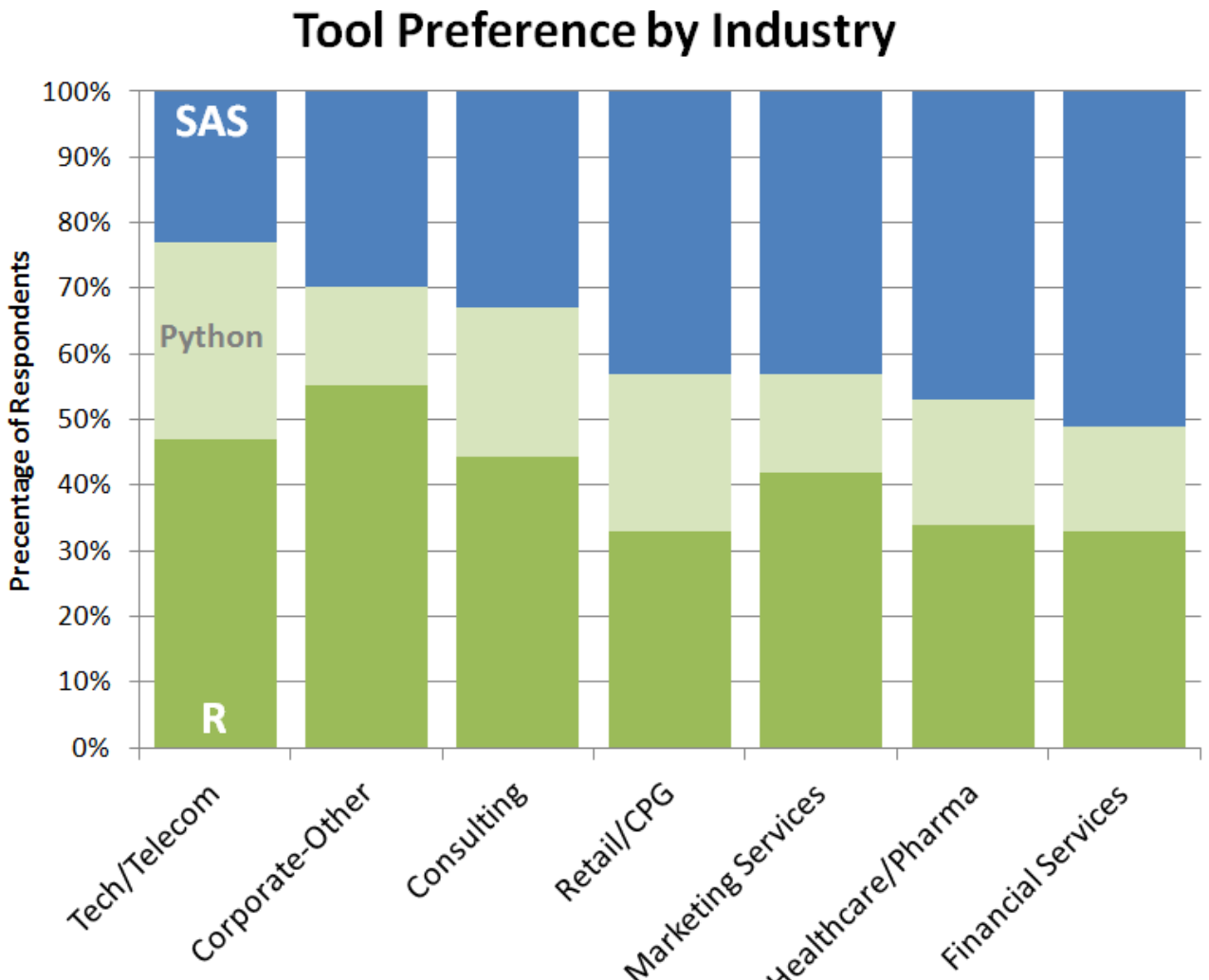
Analysis Tool	Similar Superhero	Super Powers in Common
R 	Batman 	• Detective Work • Intelligence • Cunning • Usage of Tools • More Brain than Muscles
Python 	Superman 	• Muscle Power • Super Strength • Elegance • Wide Range • More Muscles than Brain

R vs Python
KDnuggets (2016)





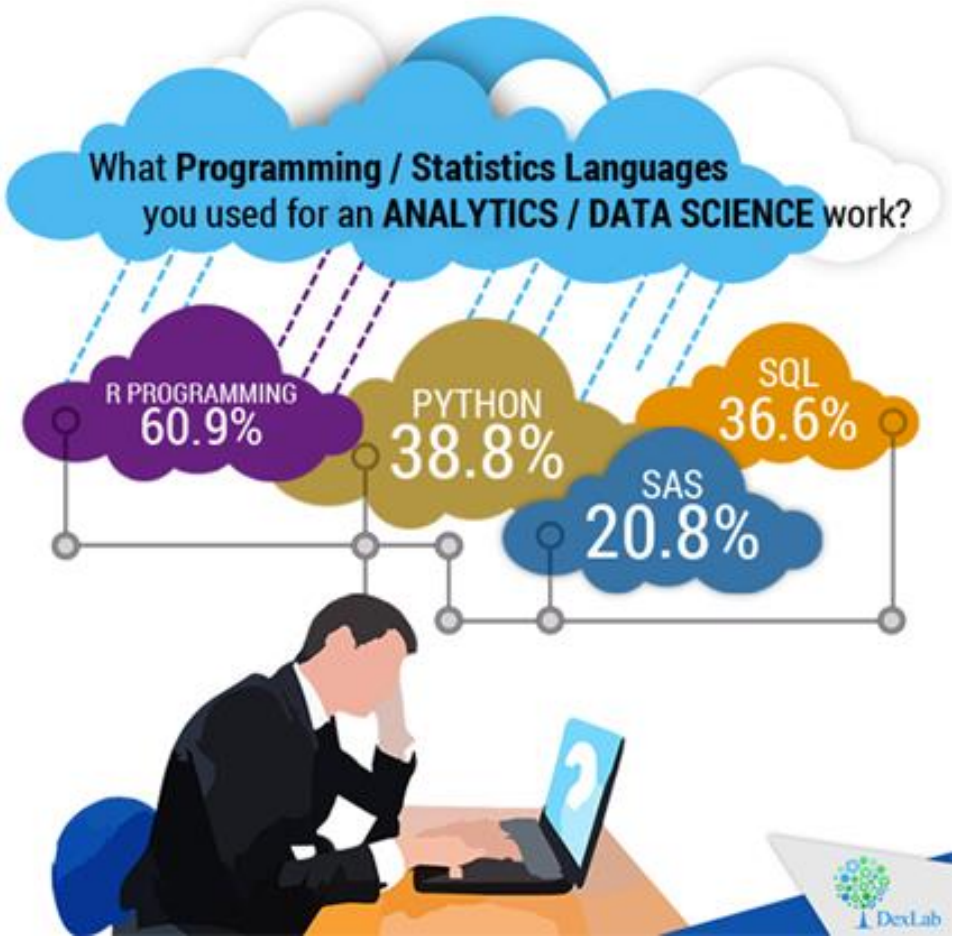
R vs Python
KDnuggets (2016)



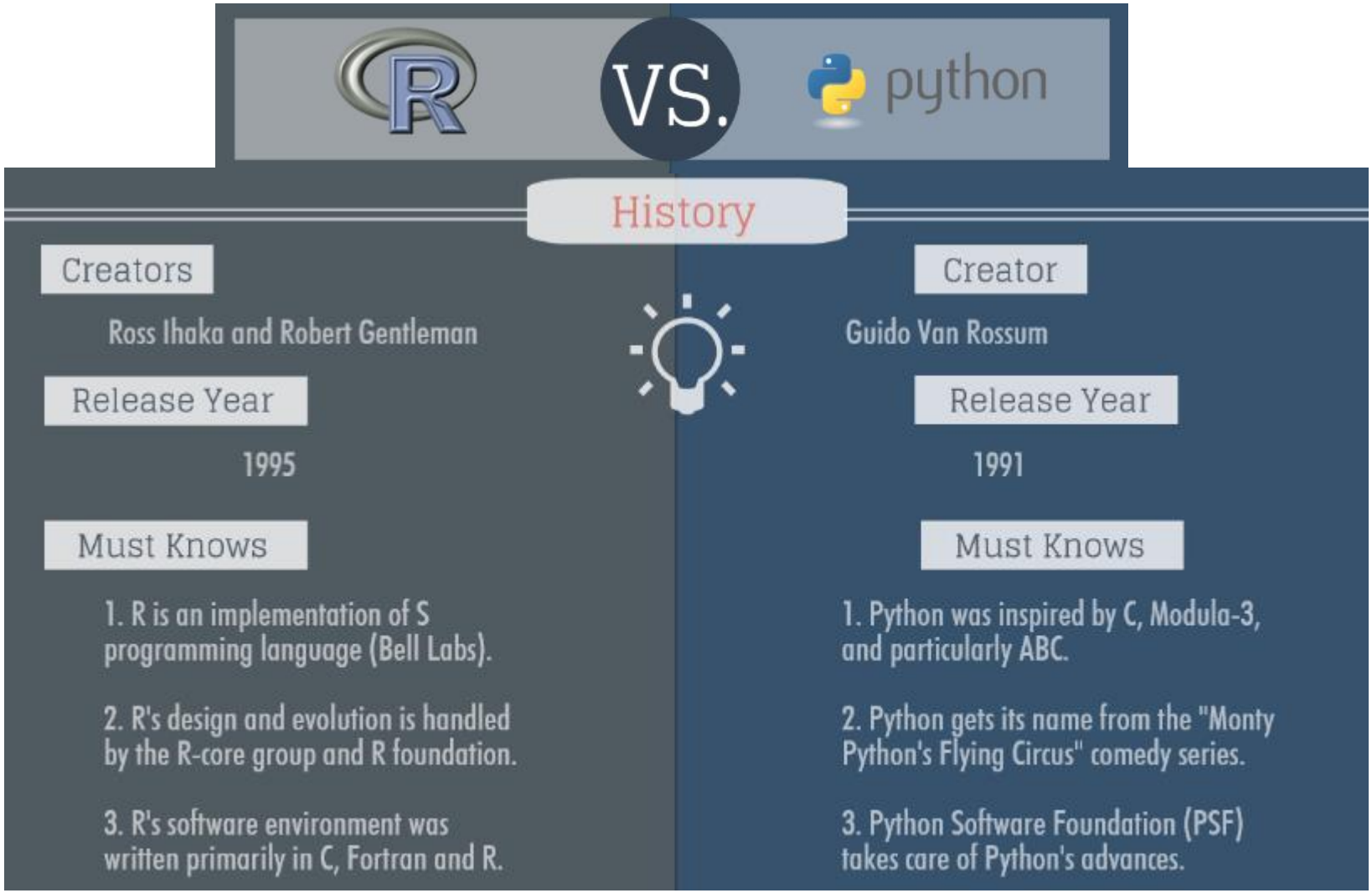
R vs Python
Analytics Vidhya (2014)

Parameter	SAS	R	Python
Availability / Cost	2	5	5
Ease of learning	4.5	2.5	3.5
Data handling capabilities	4	4	4
Graphical capabilities	3	4.5	4
Advancements in tool	4	4.5	4
Job scenario	4.5	3.5	2.5
Customer service support and Community	4	3.5	3

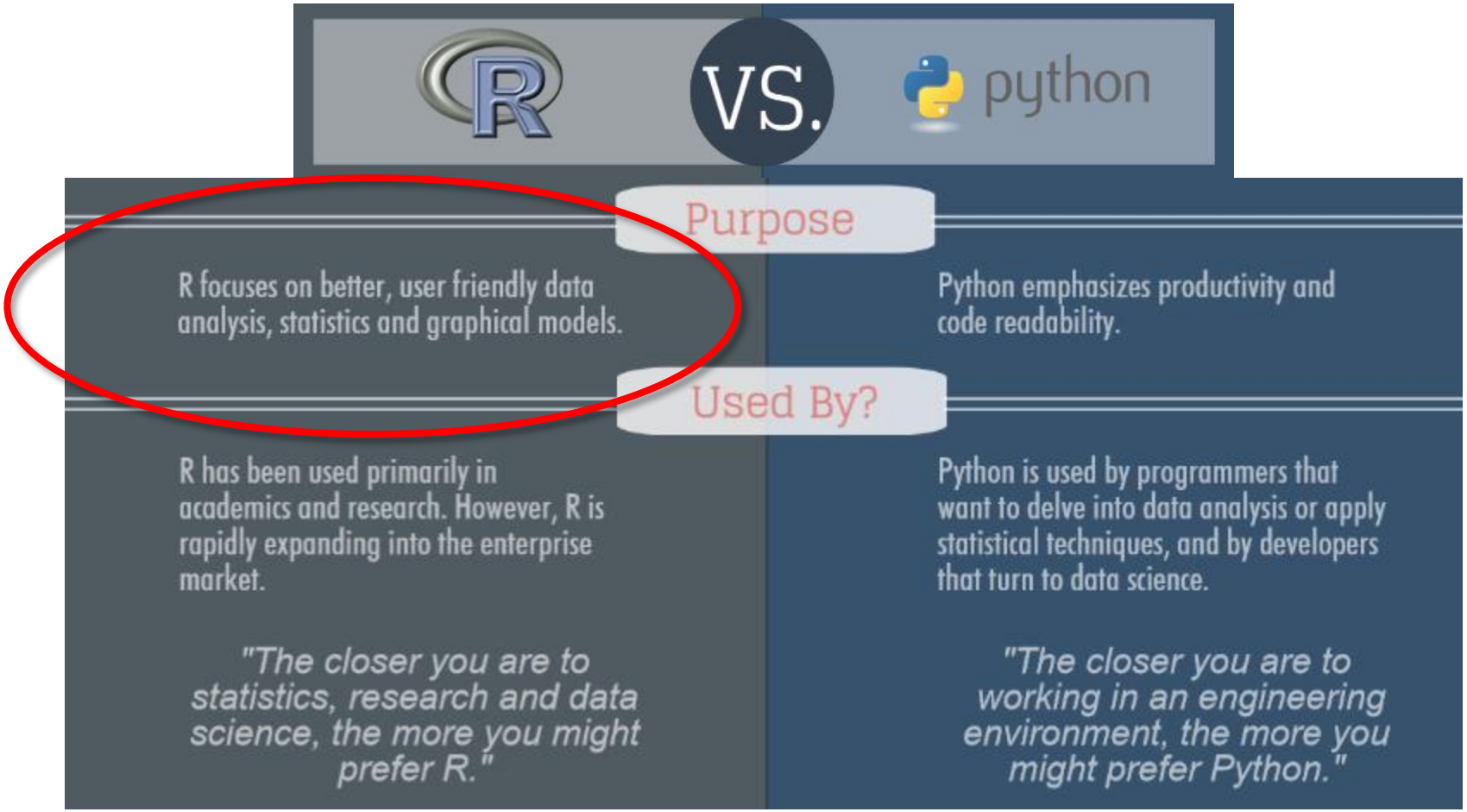
R vs Python
DexLab (2016)



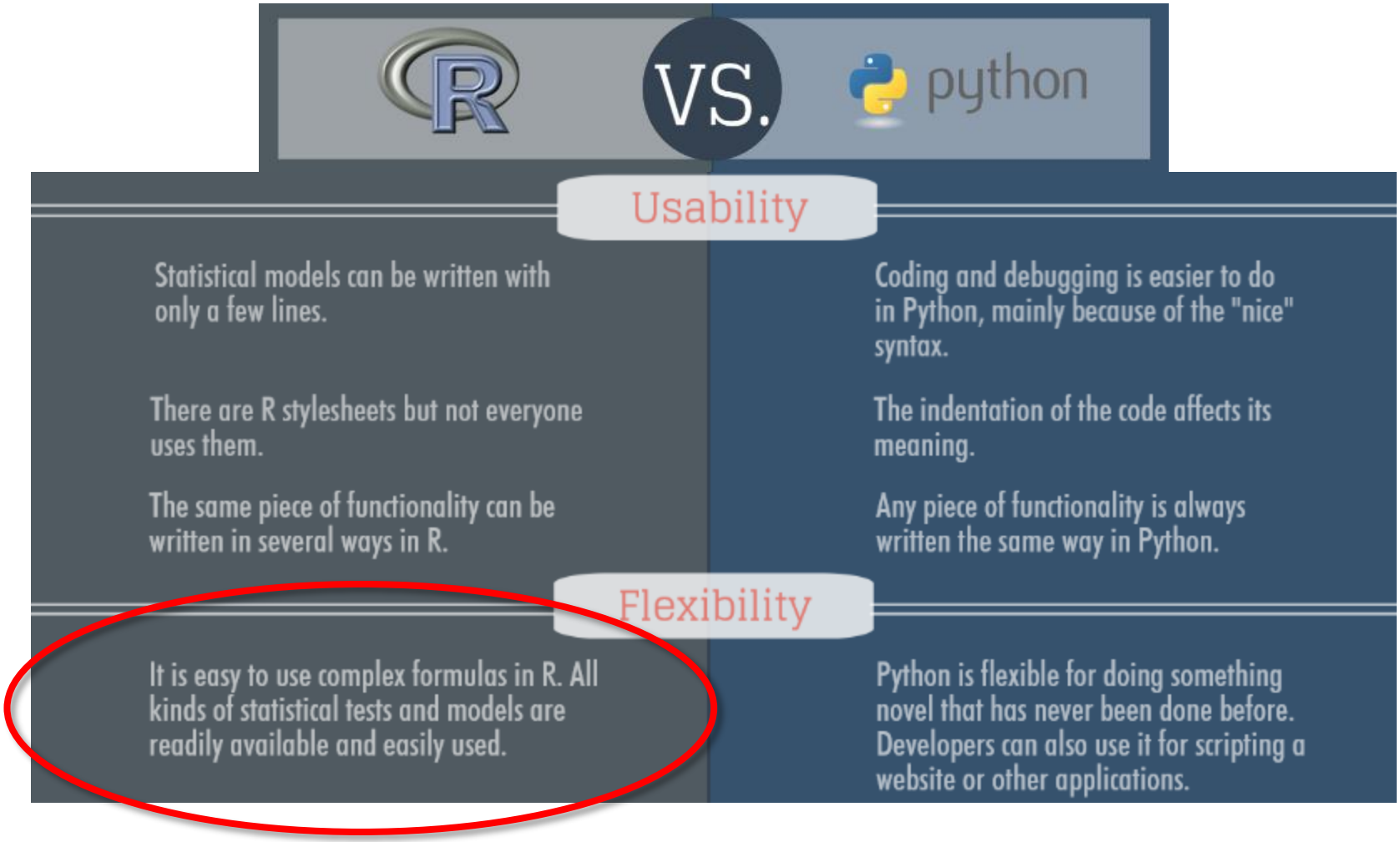
R vs Python
DataCamp: Data Science Wars (2015)



R vs Python
DataCamp: Data Science Wars (2015)



R vs Python
DataCamp: Data Science Wars (2015)



Ease of Learning

R has a steep learning curve at start. Once you know the basics, you can easily learn advanced stuff.

R is not hard for experienced programmers.

Python's focus on readability and simplicity makes that its learning curve is relatively low and gradual.

Python is considered a good language for starting programmers.

Code Repositories

CRAN stands for the Comprehensive R Archive Network: it is a huge repository of R packages to which users can easily contribute.

Packages are collections of R functions, data, and compiled code. They can be installed in R with one line.

PyPi is the Python Package Index: it is a repository of Python software, consisting of libraries. Users can contribute to Pypi, but it is a bit complicated in practice.

Watch out with dependencies and installing Python libraries!

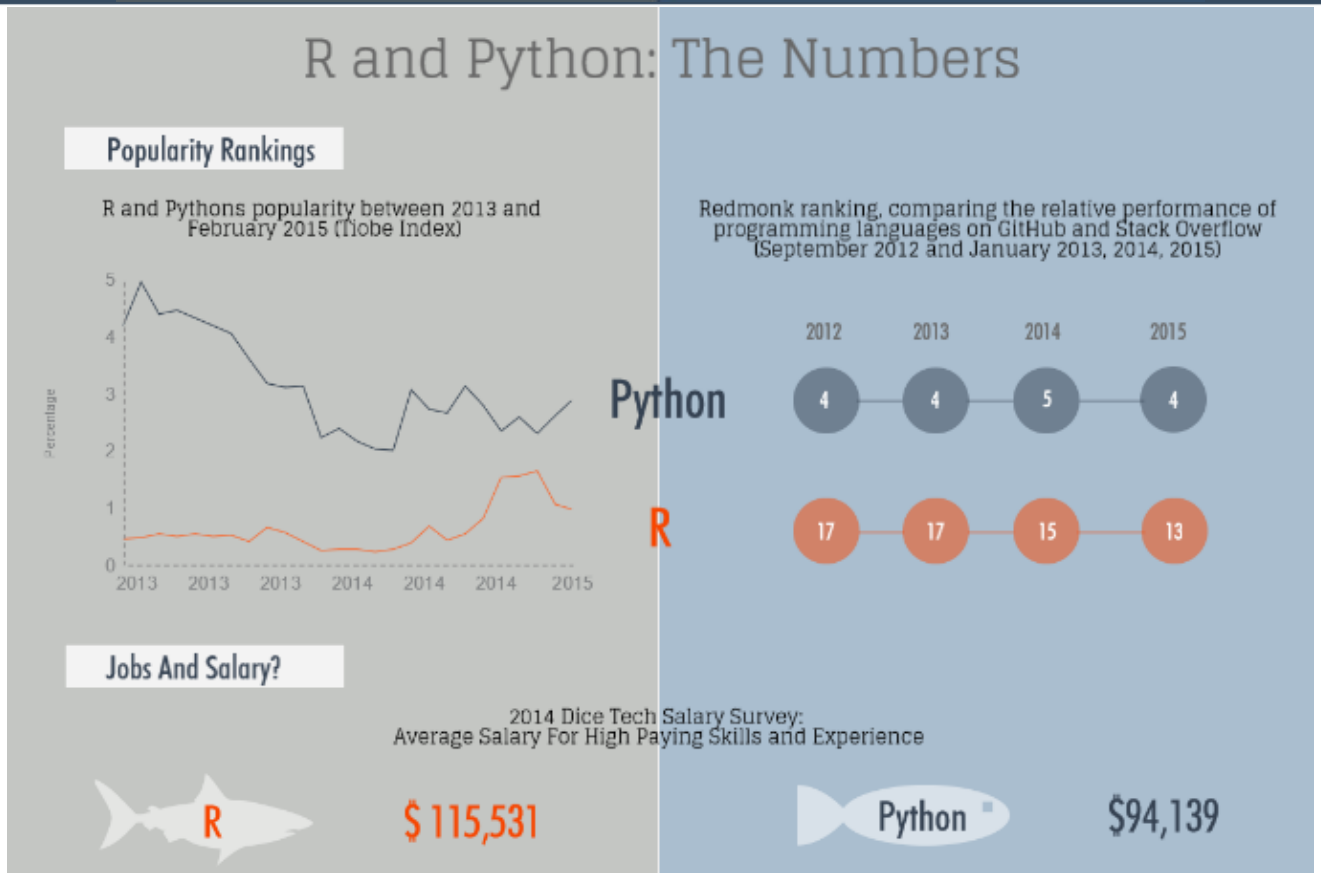
"I don't see Python [...] building up a huge code repository comparable to CRAN. [R has] a gigantic head start, [and] [...] statistics simply is not Python's central mission;"
- Norm Matloff, professor of computer science

Miscellaneous

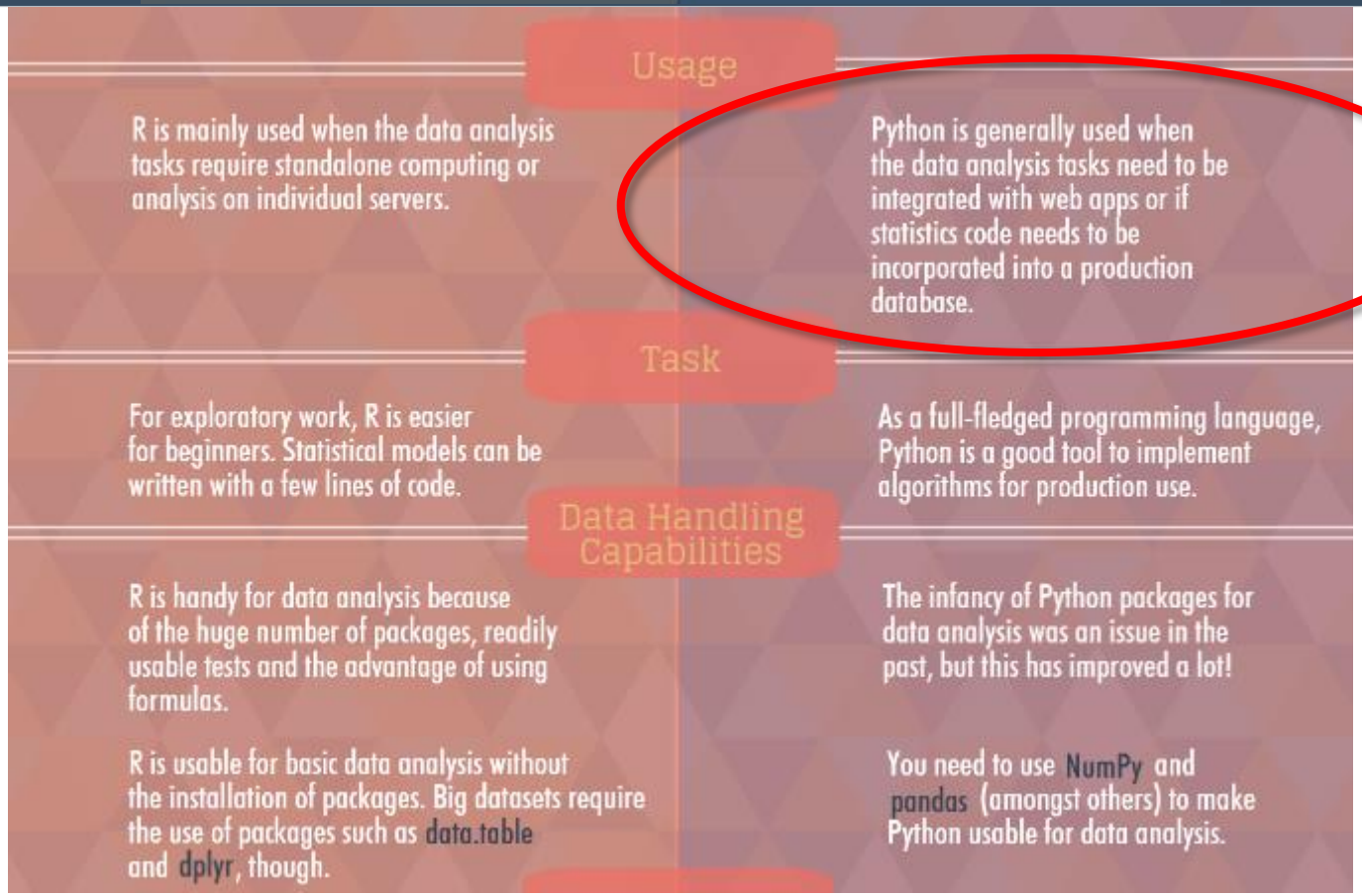
Use the rPython package to run Python code from R. Pass or get data from Python, call Python functions or methods.

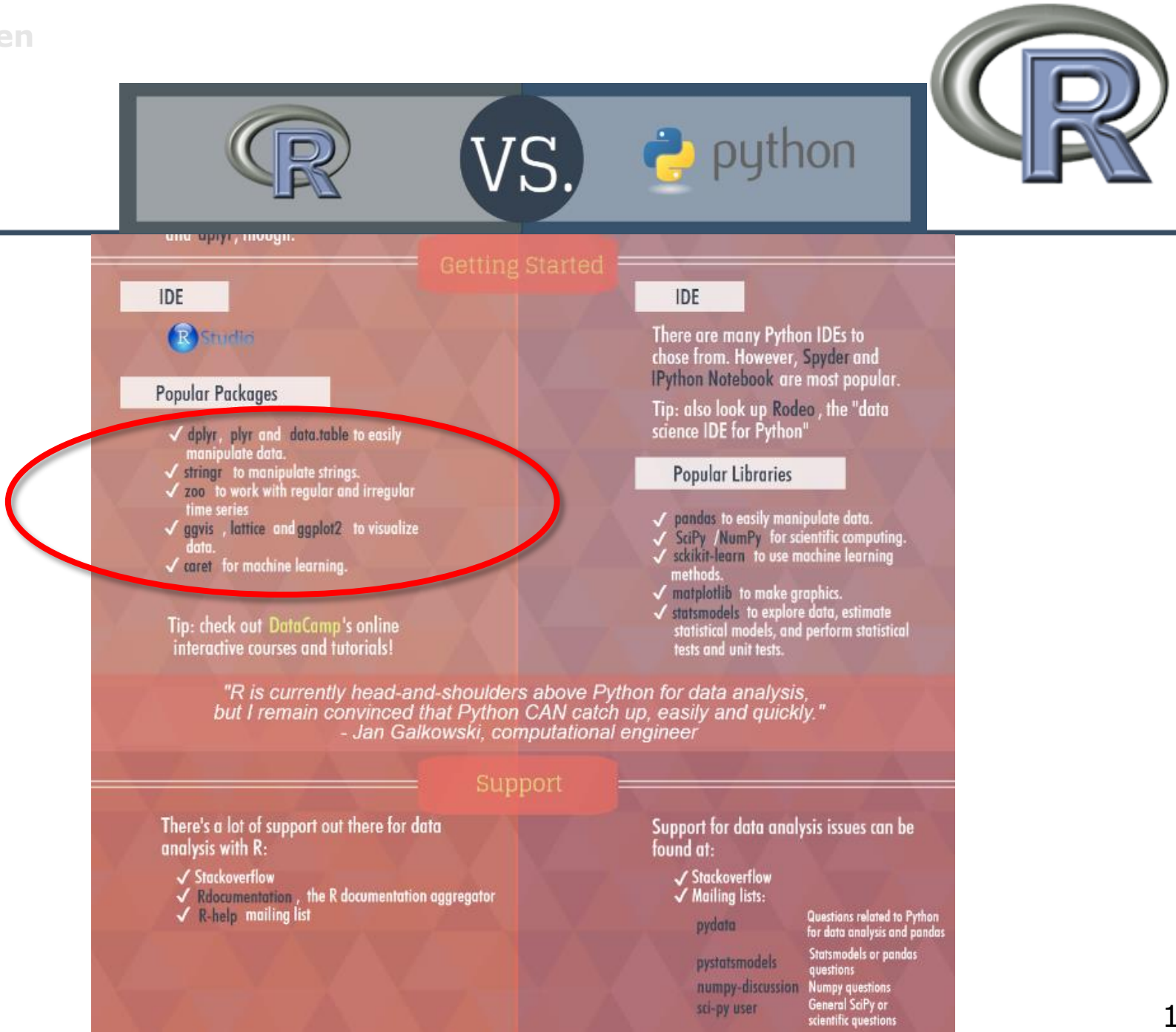
Use the RPy2 library to run R code from within Python. It provides a low-level interface from Python to R.


VS.
python

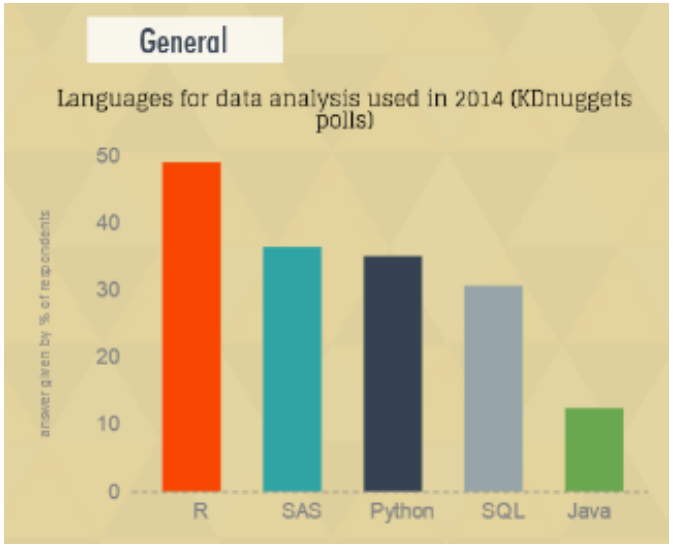

R and Python: The Numbers
Popularity Rankings
R and Pythons popularity between 2013 and February 2015 (Tiobe Index)
Percentage
5
4
3
2
1
0
2013
2013
2013
2014
2014
2014
2015
Redmonk ranking, comparing the relative performance of programming languages on GitHub and Stack Overflow (September 2012 and January 2013, 2014, 2015)
2012
2013
2014
2015
Python
4
4
5
4
R
17
17
15
13
Jobs And Salary?
2014 Dice Tech Salary Survey:
Average Salary For High Paying Skills and Experience
R
$ 115,531
Python
$94,139

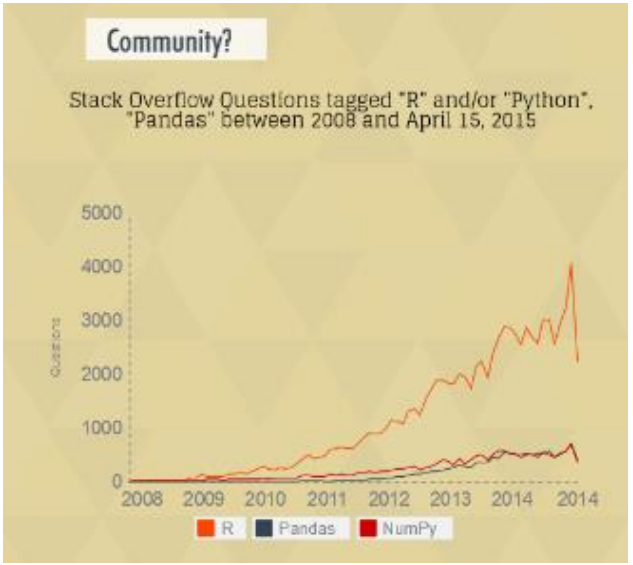

VS.
python

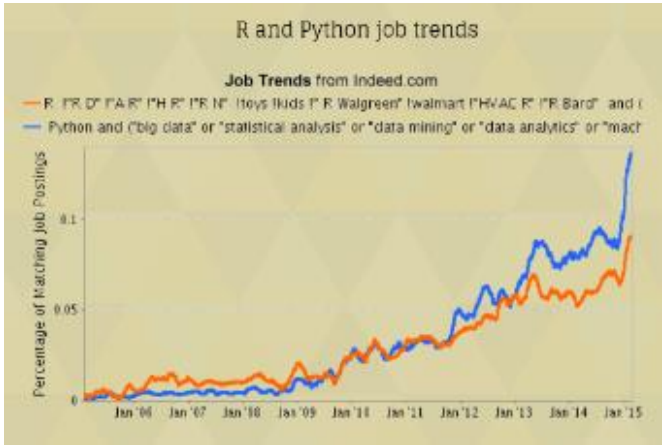

Usage
R is mainly used when the data analysis tasks require standalone computing or analysis on individual servers.
Python is generally used when the data analysis tasks need to be integrated with web apps or if statistics code needs to be incorporated into a production database.
Task
For exploratory work, R is easier for beginners. Statistical models can be written with a few lines of code.
As a full-fledged programming language, Python is a good tool to implement algorithms for production use.
Data Handling Capabilities
R is handy for data analysis because of the huge number of packages, readily usable tests and the advantage of using formulas.
The infancy of Python packages for data analysis was an issue in the past, but this has improved a lot!
R is usable for basic data analysis without the installation of packages. Big datasets require the use of packages such as data.table and dplyr, though.
You need to use NumPy and pandas (amongst others) to make Python usable for data analysis.

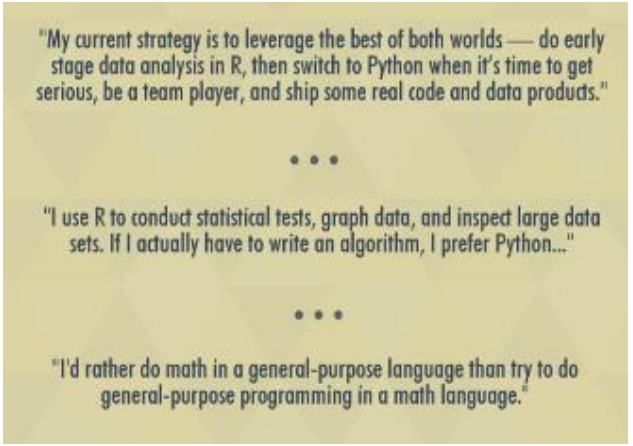

R
VS.
python
Getting Started
IDE
RStudio
Popular Packages
✓ dplyr, plyr and data.table to easily manipulate data.
✓ stringr to manipulate strings.
✓ zoo to work with regular and irregular time series
✓ ggvis, lattice and ggplot2 to visualize data.
✓ caret for machine learning.
Tip: check out DataCamp's online interactive courses and tutorials!
IDE
There are many Python IDEs to chose from. However, Spyder and IPython Notebook are most popular.
Tip: also look up Rodeo, the "data science IDE for Python"
Popular Libraries
✓ pandas to easily manipulate data.
✓ SciPy /NumPy for scientific computing.
✓ sckikit-learn to use machine learning methods.
✓ matplotlib to make graphics.
✓ statsmodels to explore data, estimate statistical models, and perform statistical tests and unit tests.
"R is currently head-and-shoulders above Python for data analysis, but I remain convinced that Python CAN catch up, easily and quickly."
- Jan Galkowski, computational engineer
Support
There's a lot of support out there for data analysis with R:
✓ Stackoverflow
✓ Rdocumentation, the R documentation aggregator
✓ R-help mailing list
Support for data analysis issues can be found at:
✓ Stackoverflow
✓ Mailing lists:
pydata
Questions related to Python for data analysis and pandas
pystatsmodels
Statsmodels or pandas questions
numpy-discussion
Numpy questions
sci-py user
General SciPy or scientific questions

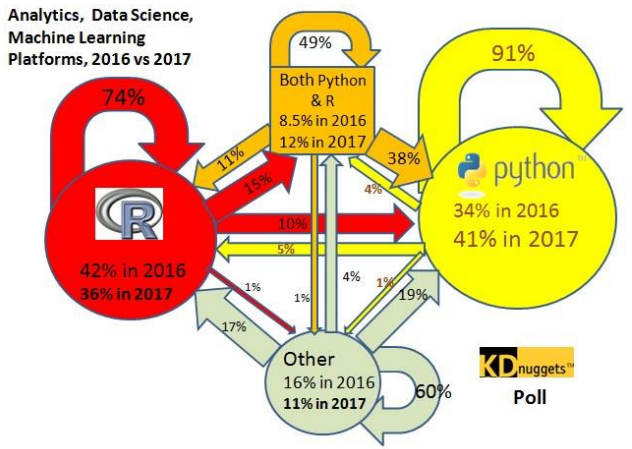
R vs Python
DataCamp: Data Science Wars (2015)











Graphical Capabilities

A picture says more than a thousand words

Visualized data can be understood more efficiently and effectively than the raw numbers alone.

R + visualization = perfect match

ggplot2	opportunity to use grammar of graphics to create layered, customizable plots
lattice	To easily display multivariate relationships
rCharts	To create, customize and publish interactive javascript visualizations from R
googleVis	To use Google Chart tools to visualize data in R
ggvis	To implement interactive grammar of graphics, while rendering in a web browser

e.g.: Visualizing Facebook friends with R

IPython Notebook

Bundle your analysis in one file

The IPython Notebook makes it easier to work with Python and data.

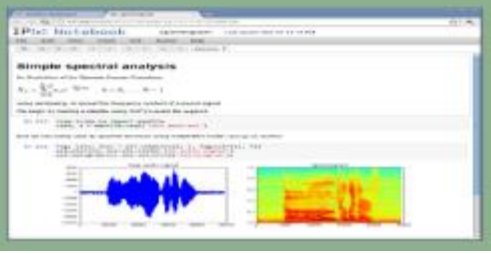
Simplify your workflow when working with data in Python

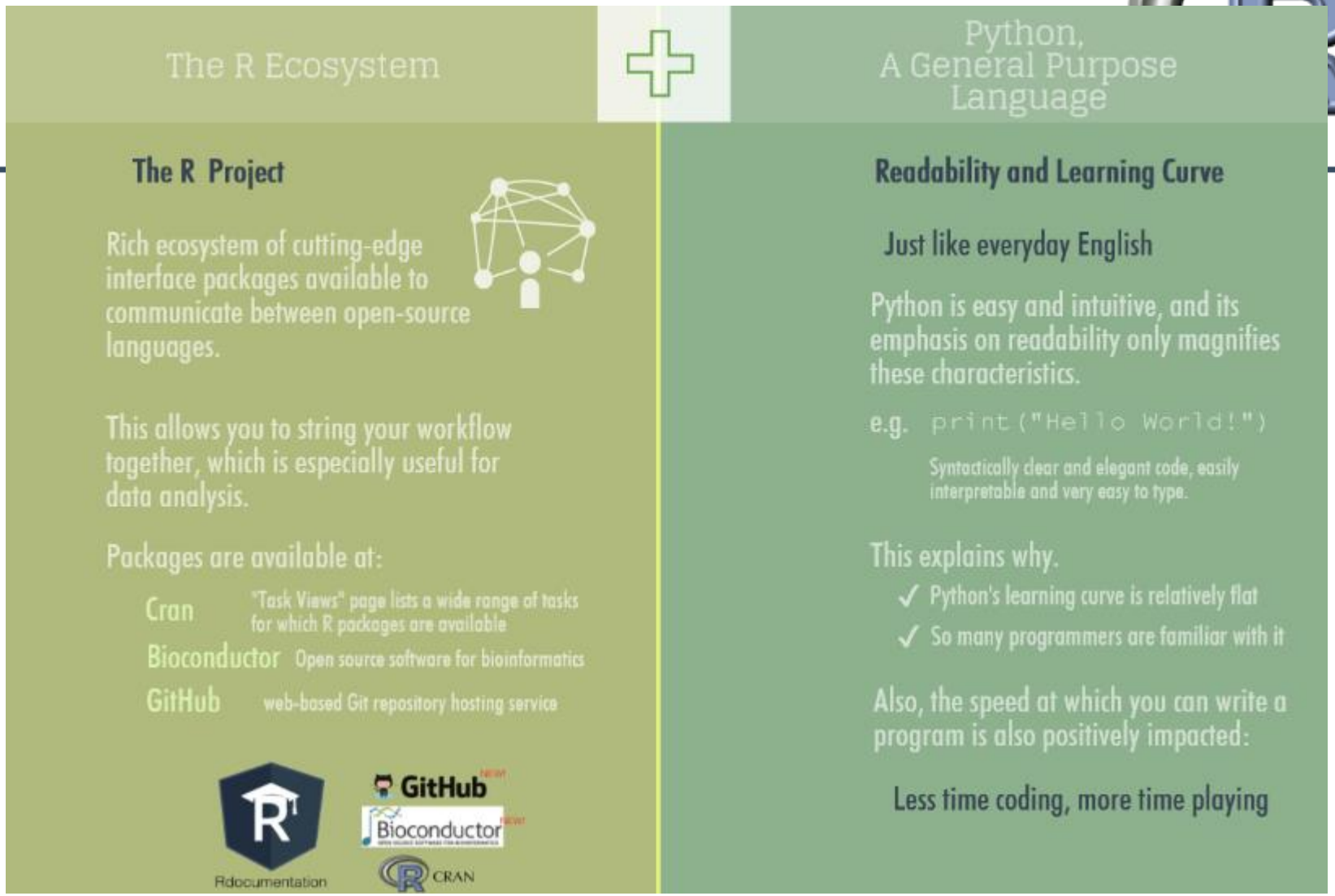
It's a combination of:

Interactive python exploration, prewritten programs, text, and equations for documentation in one environment

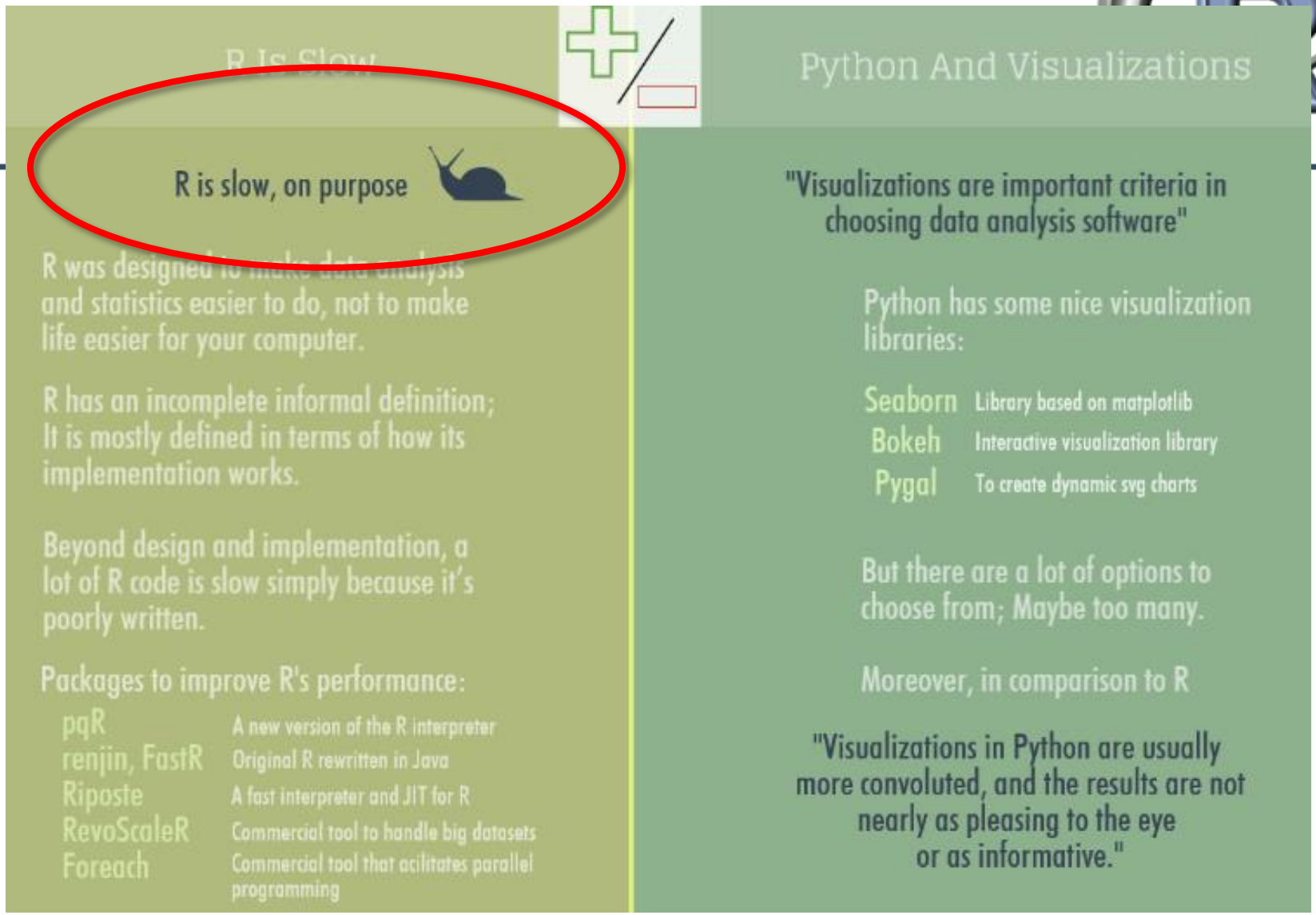
Share notebooks with colleagues without having them install anything.

The IPython notebook drastically reduces the overhead of organizing code, output, and notes files, which allows to spend more time doing real work.




The R Ecosystem
Python,
A General Purpose
Language
The R Project
Rich ecosystem of cutting-edge interface packages available to communicate between open-source languages.
This allows you to string your workflow together, which is especially useful for data analysis.
Packages are available at:
Cran "Task Views" page lists a wide range of tasks for which R packages are available
Bioconductor Open source software for bioinformatics
GitHub web-based Git repository hosting service
Rdocumentation
GitHub
Bioconductor
CRAN
Readability and Learning Curve
Just like everyday English
Python is easy and intuitive, and its emphasis on readability only magnifies these characteristics.
e.g. print("Hello World!")
Syntactically clear and elegant code, easily interpretable and very easy to type.
This explains why.
✓ Python's learning curve is relatively flat
✓ So many programmers are familiar with it
Also, the speed at which you can write a program is also positively impacted:
Less time coding, more time playing


R, Lingua Franca of Statistics
Developed by statisticians, for statisticians
Statisticians communicate ideas and methods for statistical analysis through R code and packages.
Statisticians, engineers and scientists without computer programming skills find it easy to use.
Increasing industry adoption...
R is used in finance, pharmaceuticals, media and marketing; In this last area, R's on the rise as a business analytics tool.
"The number one value to businesses in using R is access to talent"
Google
... And widespread use in academia
R is experiencing a rapid growth, solidifying its position in third place as software used in scholarly articles, right after SAS and SAP.
Python, A Multi-Purpose Language
Ready To Work!
As a common, easy-to-understand language that is known by many programmers, Python also brings people with different backgrounds together.
For example,
Some organizations that didn't want to hire or had difficulties to hire new data scientists (re)trained their existing employees to use Python instead.
This means that Python is a production ready language: it has the capacity to be a single tool that integrates with every part of your workflow!


R Is Slow
R is slow, on purpose
R was designed to make data analysis and statistics easier to do, not to make life easier for your computer.
R has an incomplete informal definition; It is mostly defined in terms of how its implementation works.
Beyond design and implementation, a lot of R code is slow simply because it's poorly written.
Packages to improve R's performance:
pqR A new version of the R interpreter
renjin, FastR Original R rewritten in Java
Riposte A fast interpreter and JIT for R
RevoScaleR Commercial tool to handle big datasets
Foreach Commercial tool that acilitates parallel programming
Python And Visualizations
"Visualizations are important criteria in choosing data analysis software"
Python has some nice visualization libraries:
Seaborn Library based on matplotlib
Bokeh Interactive visualization library
Pygal To create dynamic svg charts
But there are a lot of options to choose from; Maybe too many.
Moreover, in comparison to R
"Visualizations in Python are usually more convoluted, and the results are not nearly as pleasing to the eye or as informative."

R's Steep Learning Curve

"The worst thing about R is that ... it was developed by statisticians."

R's learning curve is nontrivial:

- Even though anybody can get results using GUIs, none is comprehensive enough to totally avoid programming.
- Finding packages can be time consuming

Using the right tools

Good resources can help you to overcome this steep learning curve:

DataCamp 's interactive exercises and tutorials

Rdocumentation to search for packages

Python Is Immature ("It's a challenger!")

A more limited way to think about data analysis

At the moment, there are no module replacements for the 100s of essential R packages

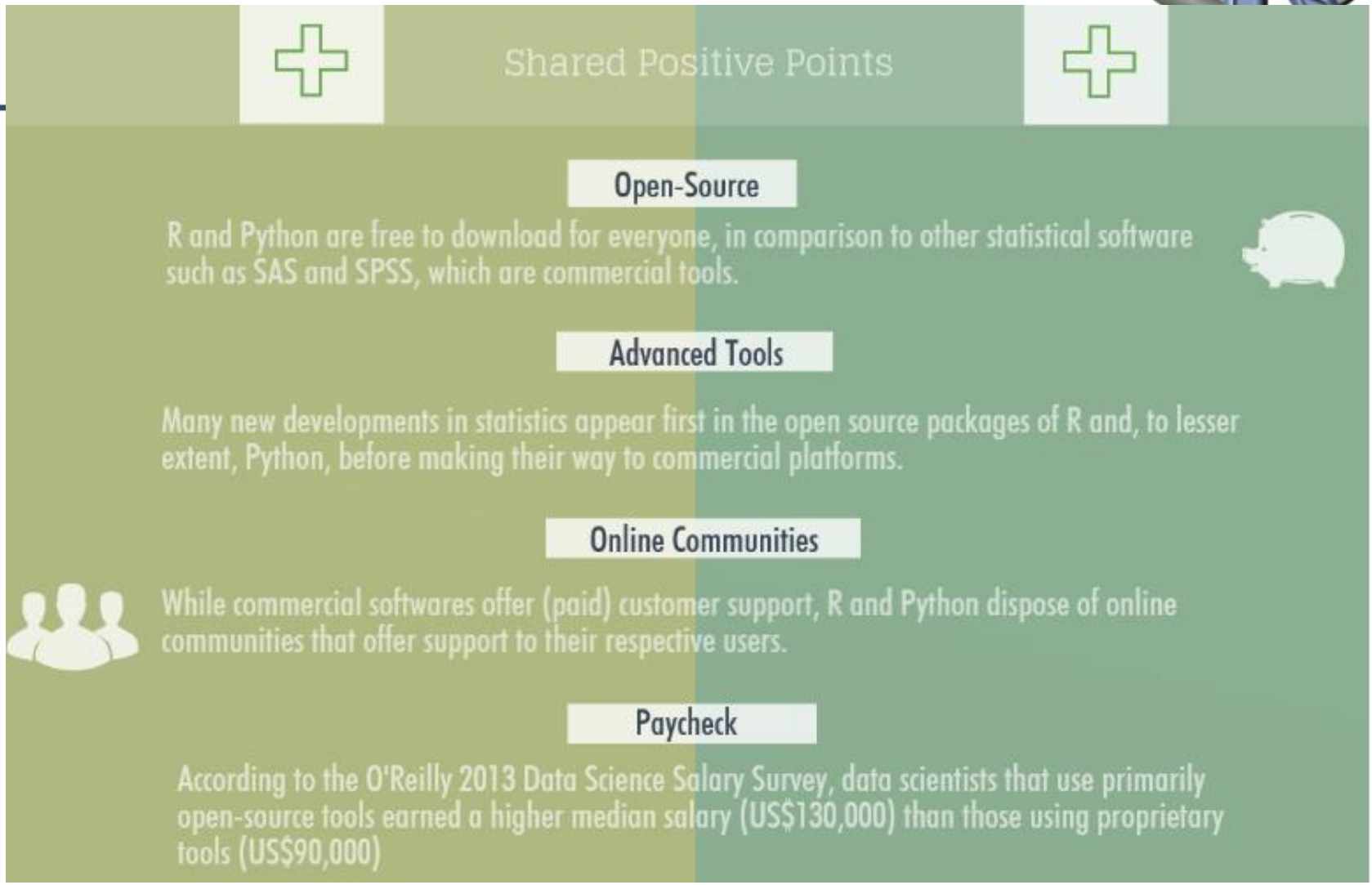
Python's catching up, but will this make people give up R?

- IPython's R extension allows you to cleanly use R in the IPython notebook.
- The current landscape of conventions and resources plays a huge role:

Matlab	Commonly used to publish open research code
Python	Used in mathematics
R	Used in statistics

Mlabwrap offers a bridge from Python to Matlab, but there are some drawbacks:

- You need to work with two languages
- You need a Matlab license


Shared Positive Points
Open-Source
R and Python are free to download for everyone, in comparison to other statistical software such as SAS and SPSS, which are commercial tools.
Advanced Tools
Many new developments in statistics appear first in the open source packages of R and, to lesser extent, Python, before making their way to commercial platforms.
Online Communities
While commercial softwares offer (paid) customer support, R and Python dispose of online communities that offer support to their respective users.
Paycheck
According to the O'Reilly 2013 Data Science Salary Survey, data scientists that use primarily open-source tools earned a higher median salary (US$130,000) than those using proprietary tools (US$90,000)

- Download unter rstudio.org
- Kostenlose IDE
  - R-Konsole
  - Editorfenster für R-Skripts
  - Plots, Hilfeseiten, Historie. Workspace-Übersicht, etc.
- RStudio als Editor
  - Befehl ausführen: `Strg + Enter` oder `Strg + R`
  - Ähnliche Befehle: `Strg + Leertaste`
  - Hilfeseiten: `F1`
- `Tools` -> `Global Options` …. -> `Pane Layout`

Hilfe
help()

- Alternativ zu `help` kann man auch `?` Verwenden

  > `help(mean)` #bzw.

  > `?mean`

- Natürlich gibt es auch Hilfe zur Hilfe

  > `help(help)` #bzw.

  > `?help`

- Sehr beliebt ist auch die Hilfeseite zu par(). Sie beschreibt eine Vielzahl von Parameters, die an Grafikfunktionen (z.B. plot()), weitergegeben werden können.

  > `help(par)` #bzw.

  > `?par`

Hilfe
help() Sonderfälle

- In einigen Fällen (arithmetische Operatoren, Kontrollanweisungen) ist es notwendig, die gesuchte Funktion in Hochkomma zu setzen.

```
# Matrixmultiplikationen
help("%*%")
?'%*%'

# Bedingte Anweisungen
help("if")
```

- Funktioniert auch, wenn eigentlich nicht nötig

```
# Zuweisung
help("mean")    # bzw.
?`mean`
```

Hilfe
help.search()

- Alternativ zu `help.search()` kann man auch `??` verwenden

  > `help.search('tree')  # bzw.`

  > `??tree`

- Öffnen und Starten der HTML-Hilfe

  > `help.start()`

- Ein Großteil der R-Hilfeseiten beinhalten einen Abschnitt mit Beispielen:

  > `example(mean)`

Hilfe
apropos()

- Die Funktion `apropos()` ist behilflich, wenn der Name der gesuchten Funktion nicht genau bekannt ist.

```
> apropos("mean")
```

- Das zurückgelieferte Ergebnis kann dann für den Aufruf der Hilfe verwendet werden

```
> help(colMeans)
```

  - colMeans(x) berechnet die Mittelwerte über die Spalten einer Datenmatrix

Hilfe
vignette() und demo()

- Einige R-Pakete enthalten so genannte Vignetten = „Handbücher“ für das jeweilige Paket.

```
> vignette(package = "zoo")    # Gibt es Vignetten
> vignette("zoo")      # Aufrufen der Vignetten
```

- Manchmal wird man erst auf der Autorenseite fündig. Für das R-Paket R Commander gibt es z.B.
  - cran.r-project.org/web/packages/Rcmdr/index.html
  - www.jstatsoft.org/v14/i09/paper
- Einige R-Pakete enthalten zudem Demos. Sie vermitteln dem Nutzer die Verwendung der beinhalteten Funktionen oder eines Teils.

```
> demo()       # listet alle verfügbaren Demos auf
> demo(graphics)  # Startet das Demo „graphics“
```

Hilfe
Links

- Offizielle Homepage R-Projekt: www.r-projekt.org
- CRAN-Netzwerk mit aktuellen Versionen: cran.r-projekt.org
- Suchmaschine von S. Goodmann: www.rseek.org
- R bloggers: www.r-bloggers.com
- Stack Overflow mit R-Filter: stackoverflow.com/questions/tagged/r
- Email-Listen
  - R-help@stat.math.ethz.ch
  - R-devel@stat.math.ethz.ch

Iris-Datensatz

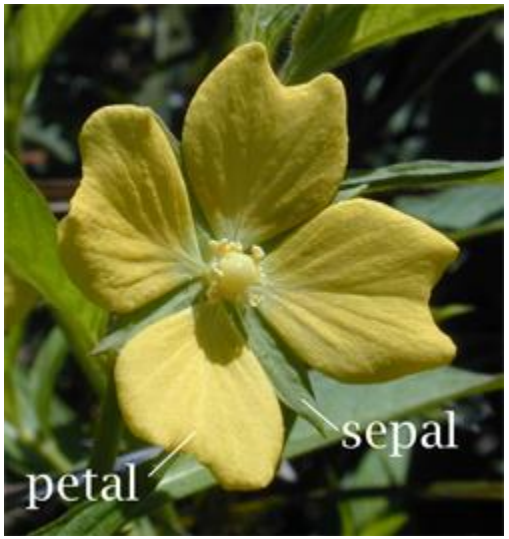
- 1936 von R. Fisher in
  „*The use of multiple measurements in taxonomic problems*"
- n = 150

Iris setosa

Iris virginica

Iris versicolor

Iris-Datensatz



- Laden des Iris-Datensatzes

```
library(datasets)
data(iris)
```

- Inhalt

```
names(iris)
```

```
[1] „Sepal.Length“ „Sepal.Width“ „Petal.Length“ „Petal.Width“ „Species“
```

  - 4 metrisch-stetig
  - 1 nominal

Iris-Datensatz

- Ansicht des Iris-Datensatzes

```
head(iris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

```
summary(iris)
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
Min. :4.300	Min. :2.000	Min. :1.000	Min. :0.100	setosa :50
1st Qu.:5.100	1st Qu.:2.800	1st Qu.:1.600	1st Qu.:0.300	versicolor:50
Median :5.800	Median :3.000	Median :4.350	Median :1.300	virginica :50
Mean :5.843	Mean :3.057	Mean :3.758	Mean :1.199	
3rd Qu.:6.400	3rd Qu.:3.300	3rd Qu.:5.100	3rd Qu.:1.800	
Max. :7.900	Max. :4.400	Max. :6.900	Max. :2.500	

Cars-Datensatz

- Autos der 30er Jahre (n = 50)
- Nur 2 Variablen: Geschwindigkeit und Bremsweg

```
names(cars)
```

```
[1] "speed" "dist"
```

```
head(cars)
```

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10

- Umfangreicherer Datensatz: Cars93 (n = 93, 27 Parameter)

```
dim(Cars93)
```

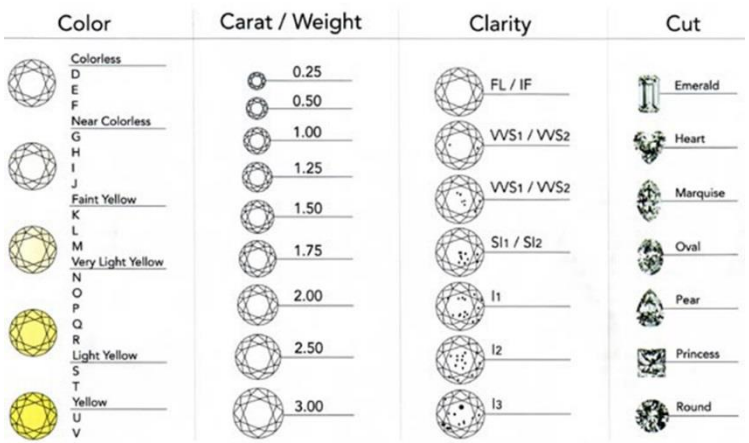
```
[1] 93  27
```

diamonds-Datensatz

- library(ggplot2)
- 10 Variablen von 53.940 geschliffenen Diamanten

`head(diamonds)`

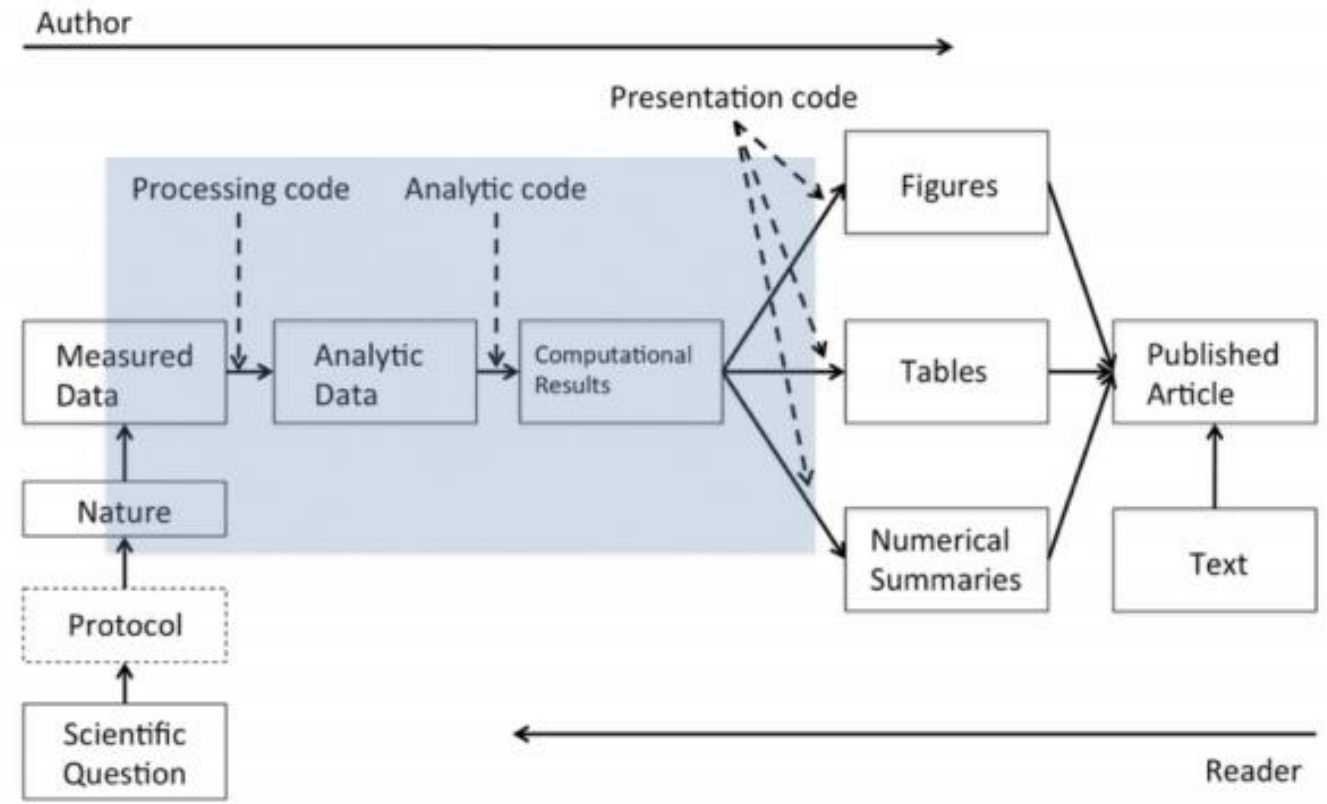
```
# A tibble: 6 x 10
  carat cut       color clarity depth table price     x     y     z
  <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1 0.230 Ideal     E     SI2      61.5   55.   326  3.95  3.98  2.43
2 0.210 Premium   E     SI1      59.8   61.   326  3.89  3.84  2.31
3 0.230 Good      E     VS1      56.9   65.   327  4.05  4.07  2.31
4 0.290 Premium   I     VS2      62.4   58.   334  4.20  4.23  2.63
5 0.310 Good      J     SI2      63.3   58.   335  4.34  4.35  2.75
6 0.240 Very Good J     VVS2     62.8   57.   336  3.94  3.96  2.48
```



R
Aufgabe 1

Probieren Sie – sofern noch nicht gemacht – die in den Folien beschriebenen Befehle. Speichern Sie den Code in einem R-Script.

Berichte
Data Science Pipeline



Peng, 2016: Report Writing for Data Science in R

Berichte
Dynamische Berichte

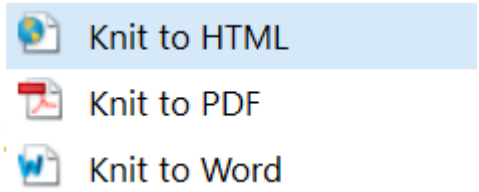
1. Analysieren der Daten, z.B. mit R
2. Einbau des Codes, der die Resultate (Tabellen, Grafiken, Zahlen) erzeugt in den Bericht
3. Erzeuge Resultate automatisch innerhalb des Codes
4. Wiederholen 1.-3. (2.+3. geschieht automatisch)

Unterstützt in R:

- Sweave (kombiniert Latex und R-Code)
- Package `knitr` (Latex/HTML/Markdown und R-Code)
  http://yihui.name/knitr

Berichte
R Markdown

- Markdown (*.md) ist eine einfache markup-Sprache mit plain text als Formatierungssyntax
- R Markdown (*.Rmd) kombiniert R Code und Markdown-Code und erlaubt das Generieren dynamischer Berichte
- R Markdown in RStudio:
  1. Öffnen eines Beispiel-Markdown-Dokuments
     `File -> New File -> R Markdown`
  2. Auswählen des Ausgabeformats (HTML, PDF, Word)
  3. Mit `Knit to …` wird der Bericht erzeugt



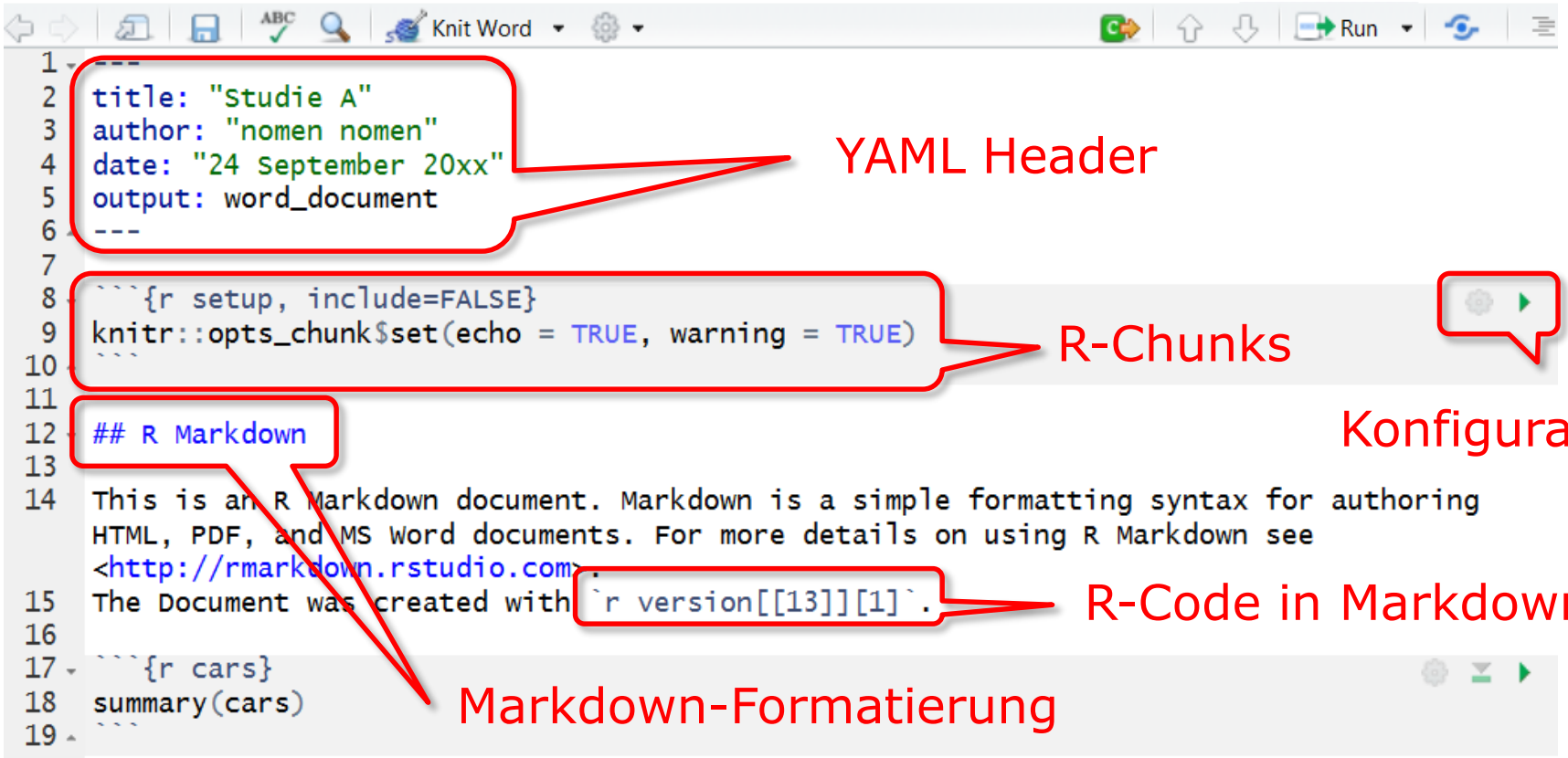
R Markdown
Aufgabe 2

- Öffnen eines R Markdown Dokuments in Rstudio

  `File -> New File -> R Markdown`

- Eingabe der Metainformation und HTML als Output-Format
- Erzeugen von Reports durch „Knit PDF / HTML / Word“
- Ändern des YAML-Headers
  - output: beamer_presentation
  - output: slidy_presentation
  - output: ioslides_presentation

R Markdown
Struktur



R Markdown Syntax

```
*italics* and _italics_

**bold** and __bold__

superscript^2^

~~strikethrough~~

[link](www.rstudio.com)

# Header 1

## Header 2

### Header 3

#### Header 4
```

italics and *italics*

bold and **bold**

superscript^2

~~strikethrough~~

link

Header 1

Header 2

Header 3

R Markdown
Listen

```
* unordered list
* item 2
    + sub-item 1
    + sub-item 2

1. ordered list
2. item 2
    + sub-item 1
    + sub-item 2
```

- unordered list
- item 2
  - sub-item 1
  - sub-item 2

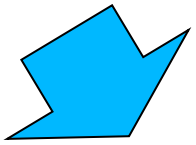
1. ordered list
2. item 2
   - sub-item 1
   - sub-item 2

R Markdown
Sonderelemente

Syntax	Ergebnis
`\*`	*
`$\epsilon$`	ϵ
`$3^{20}$`	3^{20}
`$A_b$`	A_b
`$\sum_{i=1}^{n} x_i$`	$\sum_{i=1}^{n} x_i$
`[Hyperlink](www.statistik-austria.at)`	Hyperlink
`![](path/to/image.png)`	

R Markdown
Tabellen

```
|Tabellen        |sind             |cool   |
|:---------------|:---------------:|------:|
|Spalte 1 ist    |links-bündig     |    $1 |
|Spalte 2 ist    |zentriert        |   $12 |
|Spalte 3 ist    |rechts-bündig    |  $123 |
```



Tabellen	sind	cool
Spalte 1 ist	links-bündig	$1
Spalte 2 ist	zentriert	$12
Spalte 3 ist	rechts-bündig	$123

R Markdown
Einbetten von R-Code

R Code wird in R Chunks eingebettet:

- R Code zwischen zwei Zeilen mit 3-4 backticks `
- Am Ende der ersten Zeile {r}
- Tastaturkürzel: `Strg + Alt + I`

By default

- wird der Code innerhalb der Chunks im Report ausgegeben
- wird der Code ausgeführt und der Output nach 2 hashtags ## angezeigt

Innerhalb des Textes kann R Code jederzeit zwischen `r und ` angegeben werde. Bsp.: `` `r mean(iris$Petal.Length)` ``

R Markdown
R Chunks: Optionen

Optionen innerhalb der {r} Klammern

```
```{r echo=FALSE, eval=FALSE}
mean(iris$Petal.Length)
```
```

Option	Ergebnis	Typ
`eval`	Anzeige der Ausgabe?	Logical (default: `TRUE`)
`echo`	Anzeige des R-Codes?	Logical (default: `TRUE`) oder numerisch (für best. Zeilen)
`cache`	Speichern der R-Objekte solange Chunk sich nicht ändert	Logical (default: `TRUE`)
`fig.align`	Ausrichtung von Grafiken	Character, z.B. „`center`“
`out.width`	Ändert die Grafikgröße	Character, z.B. „`60px`“, „`7cm`“
…		

R Markdown
R Chunks: echo & eval

Numerische Werte (oder Vektoren) bei echo oder eval geben an, welche Zeilen angezeigt werden sollen

````
```{r echo=2, eval=FALSE}
# this is a comment
sum(c(1,2,3))
```
````

Damit gibt der Chunk nur die Zeile

```
sum(c(1,2,3))
```

aus (kein Ergebnis, da `eval=FALSE`).

Mit eval=TRUE ergibt sich:

```
sum(c(1,2,3))
## [1] 6
```

R Markdown
R Chunks: Einbau von externem R Code

Eine R-Datei `script.R` hat folgenden Inhalt

```
# this is a comment
sum(c(1,2,3))
```

Mit der Angabe `code=readLines("script.R")` im Chunk

````
```{r echo=TRUE, eval=TRUE, code=readLines("Script.R")}
```
````

kommt man zur selben Ausgabe wie auf der letzten Seite:

```
sum(c(1,2,3))
## [1] 6
```

R Markdown
R Chunks: Child Documents

Child documents …

- erlauben es Code auszulagern bzw. das Dokument in kleinere Bereiche zu unterteilen
- erhöhen die Flexibilität (z.B. einfache Änderung der Reihenfolge)

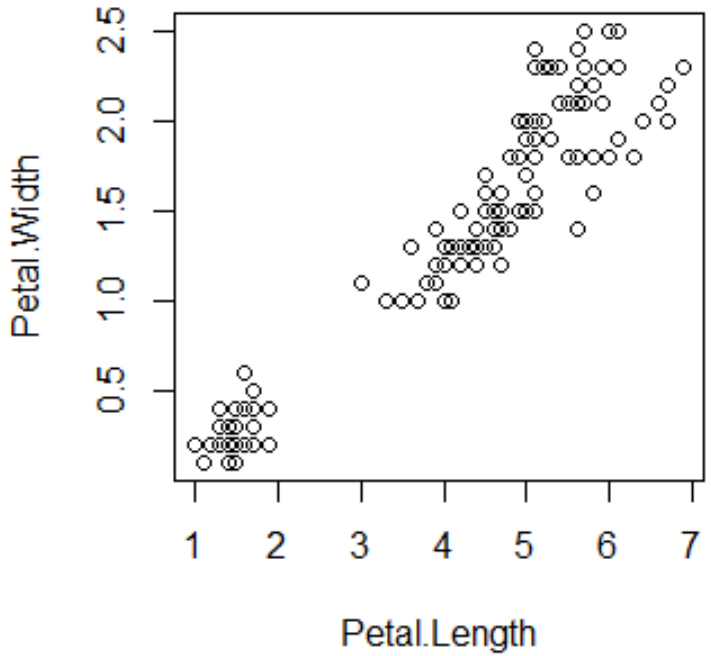
Der Einbau erfolgt allein durch Angabe des child

````
```{r child="Chapter1.Rmd"}
```
````

Cave: Keine YAML-Header im Child document

R Markdown
R Chunks: Grafiken

```
```{r  out.width=„120px“}
plot(iris[,3:4])
```
```



R Markdown
R Chunks: Tabellen

````
```{r}
iris[1:2,1:4]
```
````

Erzeugt folgenden Output:

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
```

Das knitr Package erlaubt die Erzeugung von Tabellen in Chunks mit der Funktion kable().

````
```{r}
kable(iris[1:2,1:4])
```
````

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2

R Markdown
Ausgabe: Extracting & Quick Reporting

purl() erzeugt eine Datei mit dem gesamten R-Code:

```
library(knitr)
purl("Skript.Rmd")
```

stitch() generiert aus R-Scripts schnell einen Report:

```
library(knitr)
# erzeugt eine .tex Datei (rendering nach .pdf)
stitch("file.R")
stitch_rmd("file.R") # .md Datei (rendering nach .html)
```

Das Default-Template dafür

```
system.file("misc", "knitr-template.Rmd", package="knitr")
```

Mit dem Parameter `template=„path/to/your/template.Rmd“` in stitch() kann das Template geändert werden

R Markdown
Aufgabe 3

- Einfügen in ein R Markdown Dokument
  - Satz (nach dem Chunk mit dem `label cars`), welcher die durchschnittliche (`mean`) Geschwindigkeit der Autos angibt. (Hinweis: Verwenden Sie inline R Code)
  - In einem neuen Chunk (nach dem `label cars`) sollen `dist` und `speed` in einem Streudiagramm visualisiert werden
  - Verwenden Sie die Funktion kable() um die `summary` der Cars-Daten besser darzustellen
  - Beschreiben Sie die Daten und die Ergebnisse der Analyse in dieser default-Datei. Formatieren sie den Text (highlighting) mit Markdown
- Entfernen sie in der Aufgabe oben das setup label und extrahieren Sie des gesamten R Code in eine Datei

AUFGABEN

- Für alle Aufgaben der kommenden Übungszettel müssen folgende Dateien abgegeben werden:
  - R Script
  - Markdown-Datei
  - Vollständig formatierter Bericht (inkl. Diskussion), wie ihn ein Auftraggeber bekommen würde. Enthält der Bericht auch Daten, immer mit der Funktion head(). Dazu gehören u.a. auch
    - Beschriftung bei Grafiken
    - Erklärungen gelöschter Daten, etc.
  - wenn angegeben auch die gespeicherten Daten (.RData)
- Die Aufgaben sind vorwiegend mit den Mitteln der bisher besprochenen Kapiteln zu lösen.