Cheatsheets

Czyli legalne ściąganie

Czym są cheatsheety?

Przykładowe cheatsheety

RStudio rstudio::conf Products Resources Pricing About Us Blogs

RStudio Cheat Sheets

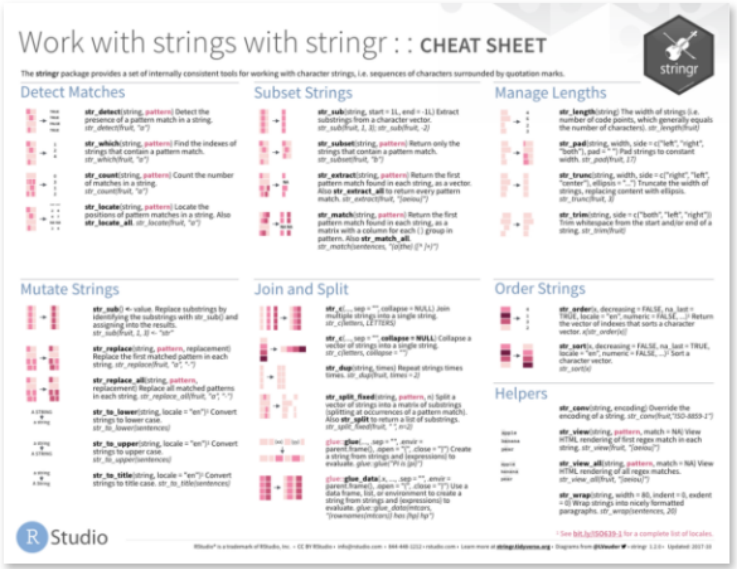
The cheat sheets below make it easy to learn about and use some of our favorite packages. From time to time, we will add new cheat sheets to the gallery. If you'd like us to drop you an email when we do, let us know by clicking the button to the right.

SUBSCRIBE TO CHEAT SHEET UPDATES HERE

Work with Strings Cheat Sheet

The stringr package provides an easy to use toolkit for working with strings, i.e. character data, in R. This cheatsheet guides you through stringr's functions for manipulating strings. The back page provides a concise reference to *regular expresssions*, a mini-language for describing, finding, and matching patterns in strings. Updated 10/17.

DOWNLOAD



https://www.rstudio.com/resources/cheatsheets/

Dlaczego warto je tworzyć, po co są nam one potrzebne w naszej grupie?

Nasze cheatsheety

factorMerger
Cheat Sheet

Agnieszka Sitko [aut, cre]
Przemysław Biecek [aut, ths]
University of Warsaw

Introduction

How to check if averages are different among k groups? Use **ANOVA!**

How to visualise how these groups are different? Use **factorMerger!**

The aim of **factorMerger** is to provide informative and easy to understand visualisations of *post-hoc* comparisons. It gives consistent and non-overlapping adaptive fusing of groups based on likelihood ratio test (LRT). The package **factorMerger** works for wide spectrum of families like Gaussian, binomial or survival.

Results from the adaptive fusing are presented with the *Merging Paths Plots* - a hierarchical representation of LRT-based distances among groups.

In addition, the *Generalized Information Criterion (GIC)* is presented for fused models. This criterion may be used to choose the optimal segmentation of groups.

Graphical summary of the variable of interest in each group is presented in the right panel.

Find more in *https://arxiv.org/abs/1709.04412*

Example

```
library(factorMerger)

fmAll <- mergeFactors(
    response  = pisaEuro$math,
    factor = pisaEuro$country,
    method = "fast-adaptive",
    family = "gaussian")

print(fmAll)

plot(fmAll,
    panel = "all",
    responsePanel = "tukey")
```

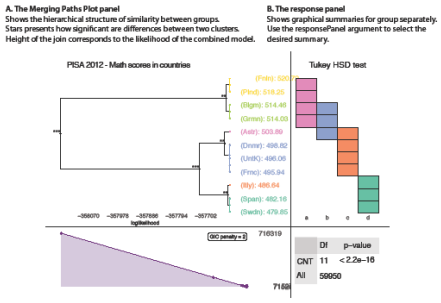
factorMerger in version 0.3.2 (2017) Agnieszka Sitko, Przemysław Biecek https://CRAN.R-project.org/package=randomForestExplainer - CC BY

Set of tools to support post-hoc segmentation of groups

Four panels to see it all

A. The Merging Paths Plot panel
Shows the hierarchical structure of similarity between groups. Stars presents how significant are differences between two clusters. Height of the join corresponds to the likelihood of the combined model.

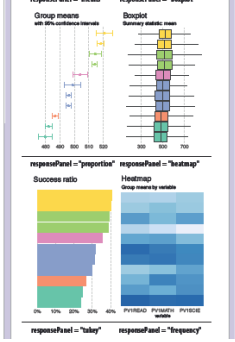
B. The response panel
Shows graphical summaries for group separately. Use the responsePanel argument to select the desired summary.



C. The GIC panel
Shows the *Generalized Information Criterion* for all models in MPP panel. Default *penalty*=2 corresponds to the AIC criterion. Colours in the MPP panel corresponds to the optimal model.

D. The summary panel
Shows the no. groups, no. observations, and the p-value for global hypothesis

The response panel summaries



Key arguments of

The **factorMerger** package is highly customisable.
The **panel** argument selects panels to be plotted.
The **nodesSpacing** argument selects how MPP nodes shall be spaced.
The **responsePanel** argument selects which summary shall be presented in panel B.
The **responsePanelPalette** argument selects the colouring palette.
The **colorClusters** argument selects if segments of groups shall be colour coded.
The **penalty** argument selects penalty for GIC criterion.

Argument	Summary
panel = "all"	All panels
panel = "response"	Panels A and B
panel = "GIC"	Panels A and C
panel = "tree"	Only panel A

randomForestExplainer
What's in the forest?

Aleksandra Paluszynska [aut, cre]
Przemysław Biecek [aut, ths]
University of Warsaw

Basics

The aim of the **randomForestExplainer** package is to support structure exploration and visualisation for a random forest model.

Once you have a model created with the **randomForest** package, use following functions to examine its structure.

```
library(randomForest)
library(randomForestExplainer)

forest <- randomForest(PV1MATH~.,
data = pisa2015, localImp = TRUE)
```

Variable Importance

The **measure_importance()** function calculates different measures of importance for variables presented in the forest. Note that different variables are available for classification forests and regression forests.

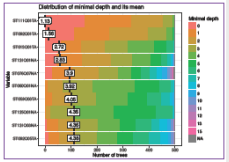
randomForestExplainer - Structure mining and visualisation for Random Forests

Variable Depth

The **min_depth_distribution()** function calculates distribution of minimal depth of given variable in all trees. Use the **plot_min_depth_distribution()** function to plot this distribution along with mean depths for variables. In general, the higher are variables the more influential they are.

```
forest_frame <-
min_depth_distribution(forest)

plot_min_depth_distribution(forest_fram
e)
```



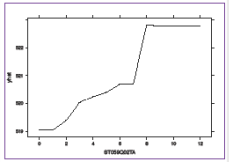
The **min_depth_interactions()** function

Partial Dependence Plots

The **partial()** function from the **pdp** package calculates marginal relation between target variable and selected one or two independent variables. The relation can be noted with **lattice** graphical system with the **plotPartial()** function or with **ggplot2** system with the **autoplot()** function.

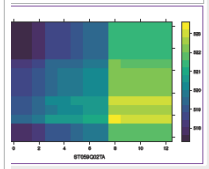
```
library(pdp)

pdp1 <- partial(forest, "ST0059Q02TA")
plotPartial(pdp1)
```



Variable depth and variable importance ions are useful in identification which ble/variables are worth watching, while the plots are useful to understand the nature of relation between target variable and ble/s of interest.

```
<- partial(forest, c("ST0059Q02TA",
S1Q03NA"))
Partial(pdp2)
```



Local Approximations

Based on LIME

Literature

ggRandomForests: Random Forests for Regression. John Ehrlinger (2016)

pdp: An R Package for Constructing Partial Dependence Plots. Brandon M. Greenwell (2017)

forestFloor: Forest Floor Visualizations of Random Forests. Soeren Welling, Hanne Refsgaard, Per Brockhoff, Line Clemmensen (2016)

MLExpResso
Cheat Sheet

Aleksandra Dąbrowska [aut, cre]
Alicja Gosiewska [aut]
Przemysław Biecek [aut, ths]

Introduction

MLExpResso is an R package for integrative analyses and visualization of gene expression and DNA methylation data.

Key functions of this package are:

- identification of genes with affected expression – **calculate_test()** function,
- identification of DMR - differentially methylated regions - **calculate_test()** function,
- identification of regions with changes in expression and methylation - **calculate_comparition_table()** function,
- visualization of identified regions – **plot_gene()** and **plot_volcanoes()** functions.

The joint modeling and visualization of genes expression and methylation improve interpretability of identified signals.

Identification of genes with affected expression

MLExpResso::calculate_test(data, condition, test)

Function calculate_test() computes log folds, p-values and means for chosen test for both, methylation and expression data.

```
> BRCA_exp[1:3, 1:5]
```

	SUBTYPE	AANAT	AARSD1	AATF	AATK
TCGA-A1-A0SB-01A-11R-A144-07	Normal	9	2354	2870	317
TCGA-A1-A0SD-01A-11R-A115-07	LumA	2	1846	5656	312
TCGA-A1-A0SF-01A-11R-A084-07	LumA	11	3391	9522	736

Argument test allows using many different statistic tests for finding differences in expression. All available values are in the table below.

Value	Test
'ttest'	student's t-tests
'nbinom2'	negative binomial tests
'lrt'	likelihood-ratio test
'qlf'	quasi-likelihood F-test

Example

```
library("MLExpResso")
library("MLExpRessoData")
exp <- BRCA_exp[ ,-1]
gr_exp <- BRCA_exp$SUBTYPE
gr_exp <- ifelse(gr_exp == 'LumA', 'LumA', 'other')
res_exp <- calculate_test(exp, gr_exp, 'lrt')

> head(res_exp)
```

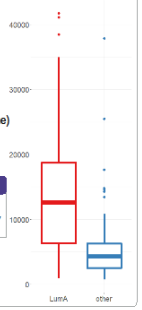
	id	log2.fold	pval	mean_LumA	mean_other	mean
1	AURKB	2.3	3.2e-32	539	2324	1485
2	CBX2	2.9	2.8e-26	633	4297	2574
3	KDNAJ	1.4	8.6e-24	11547	26427	19444
4	PKK11	3.8	2.3e-22	396	3480	2031
5	BIRC5	2.0	2.0e-21	1957	6658	4119
6	GSG2	1.4	4.5e-21	278	629	464

MLExpResso::plot_diff_boxplot(data, condition, gene)

Function plot_diff_boxplot() generates a boxplot of values from choosen data frame column with division in groups (two or more).

Example

```
plot_diff_boxplot(data = exp,
                  condition = gr_exp,
                  gene = 'CACNA1G')
```



MLExpRessoData

The methodology is supplemented with example applications to The Cancer Genome Atlas data.

MLExpRssoData is an R package which contains information from *The Cancer Genome Atlas (TCGA)* Data Portal. Data sets in this package are based on Bioconductor package **RTCGA**. In examples, we use both, methylation and expression data.

- **BRCA_exp** - It contains information about gene expression: read counts per-gene, computed for genes for 736 patients with breast cancer. Rows of this data set correspond to samples taken from patients. First column *SUBTYPE* corresponds to a subtype of BRCA cancer, next columns correspond to genes.
- **BRCA_met** - It contains information about methylation of CpG probes for patients with breast cancer. Rows of this data set correspond to patients, more precisely, to samples taken from patients. First column *SUBTYPE* corresponds to a subtype of BRCA cancer, next columns correspond to CpG probes. Values inside the table indicate the percentage methylation level of CpG probe for a specified sample.

For aggregation CpG probes to correspond genes we use the *Illumina human methylation* data set from **TxDb.Hsapiens.UCSC.hg18.knownGene** Bioconductor package.

Identification of DMR - differentially methylated regions

MLExpResso::calculate_test(data, condition, test)

Argument test allows using two different statistic tests for finding differences in methylation levels. All available values are in the table on the right.

Value	Test
'ttest'	student's t-test
'methyanalysis'	quasi-likelihood F-test

MLExpResso::aggregate_probes(data)

Function aggregate_probes() aggregates CpG probes to corresponding genes using, by default, *the Illumina human methylation* data.

```
> BRCA_met[1:3, 1:5]
```

	SUBTYPE	cg00021537	cg00031163	cg00033227	cg00050313
TCGA-A1-A0SD-01A-11D-A117-05	LumA	0.018	0.71	0.3064	0.024
TCGA-A2-A04N-01A-11D-A112-05	LumA	0.014	0.74	0.3088	0.028
TCGA-A7-A0AF-01A-11D-A012-05	Basal	0.014	0.70	0.3094	0.014

Example

```
met <- aggregate_probes(BRCA_met)
gr_met <- BRCA_met$SUBTYPE
gr_met <- ifelse(gr_met == 'LumA', 'LumA', 'other')
res_met <- calculate_test(met, gr_met, 'ttest')
> head(res_met)
```

	id	log2.fold	pval	mean_LumA	mean_other	mean
1	ICAM2	-0.152	3.8e-17	0.25	0.41	0.33
2	RILP	0.051	2.6e-13	0.41	0.36	0.38
3	PTPDC	0.115	5.4e-12	0.47	0.31	0.36
4	TNFSF12	-0.134	5.9e-12	0.18	0.31	0.25
5	CD7	0.098	1.6e-11	0.86	0.77	0.81
6	KSR1	0.200	2.1e-11	0.66	0.46	0.55

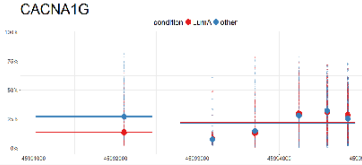
MLExpResso::plot_methylation_path(data, condition, gene)

Function plot_methylation_path() visualizes a chosen gene with marked CpG probes.
Y axis describes methylation level.
X axis describes a location of the probe on the chromosome.
Horizontal lines show the mean methylation level for each Island in a division to groups.
Groups are defined by colors. Large dots symbolize means of methylation level for CpG probes, small dots symbolize methylation levels for each observation.

Example

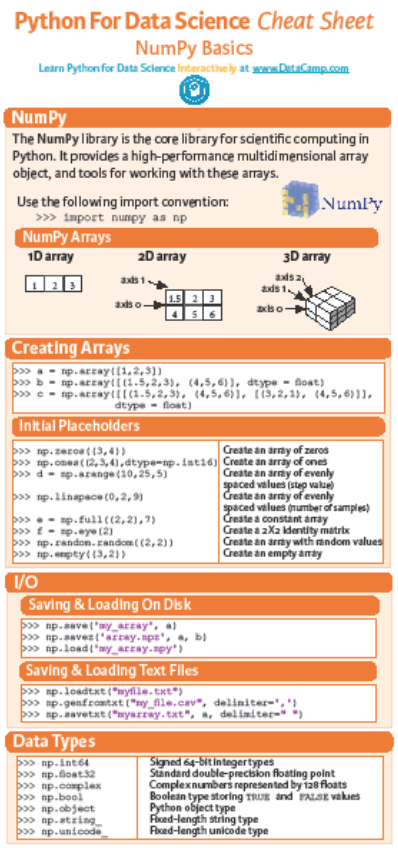
```
plot_methylation_path(data = BRCA_met ,
                      condition = gr_met,
                      gene = 'CACNA1G',
                      observ = T)
```

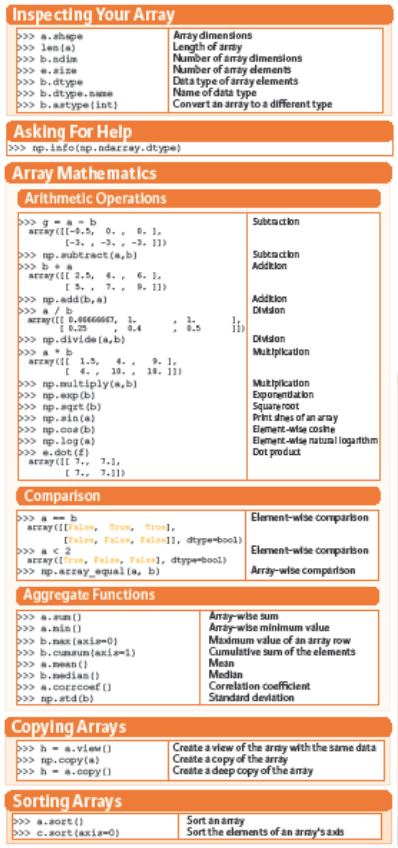
CACNA1G

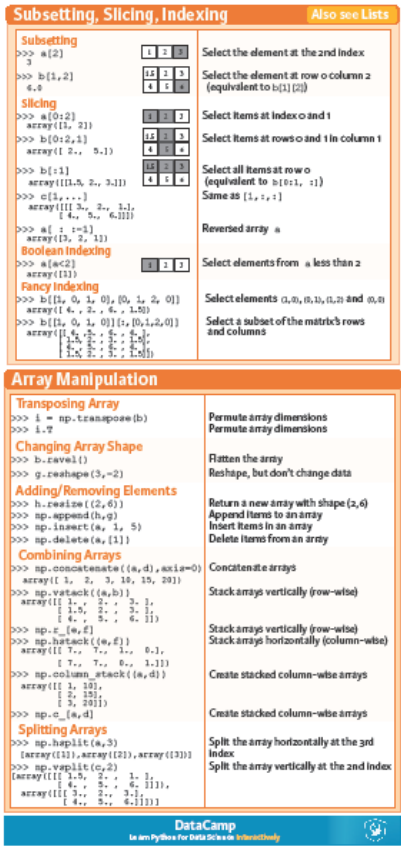


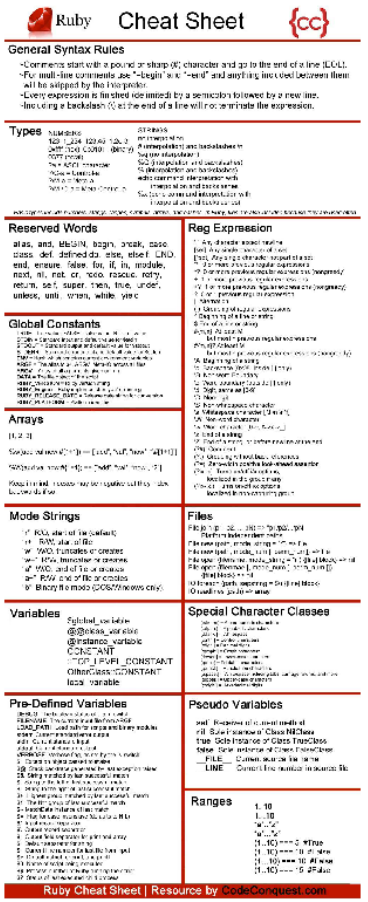
MLExpResso in vesrion 0.2.0(2017) Aleksandra Dąbrowska, Alicja Gosiewska, Przemysław Biecek - CC BY

Nie tylko R









https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Numpy_Python_Cheat_Sheet.pdf

http://www.codeconquest.com/wp-content/uploads/Ruby-Cheat-Sheet-by-CodeConquestDOTcom.pdf

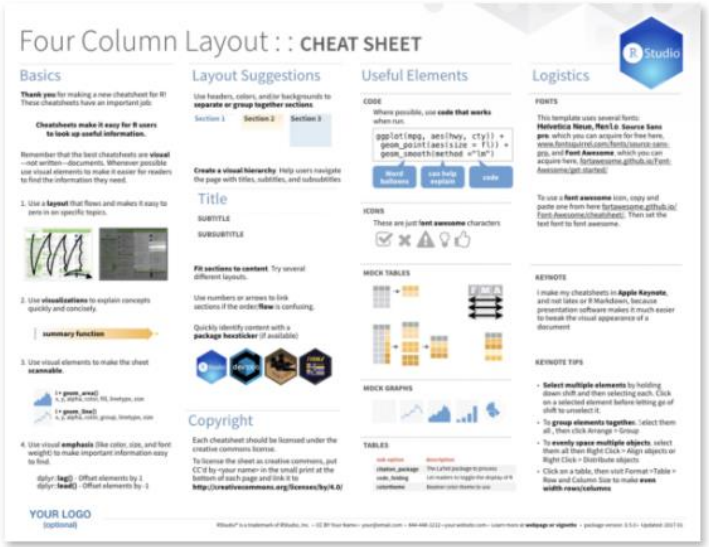
MI

Jak je zrobić?

RStudio rstudio::conf Products Resources Pricing About Us Blogs

Want to contribute a cheatsheet of your own?

We'd like to help you make and share high quality cheatsheets on R topics. The template below provides a useful starting place. It contains tips for designing a three or four column cheatsheet, as well as reusable elements to build your sheet with.

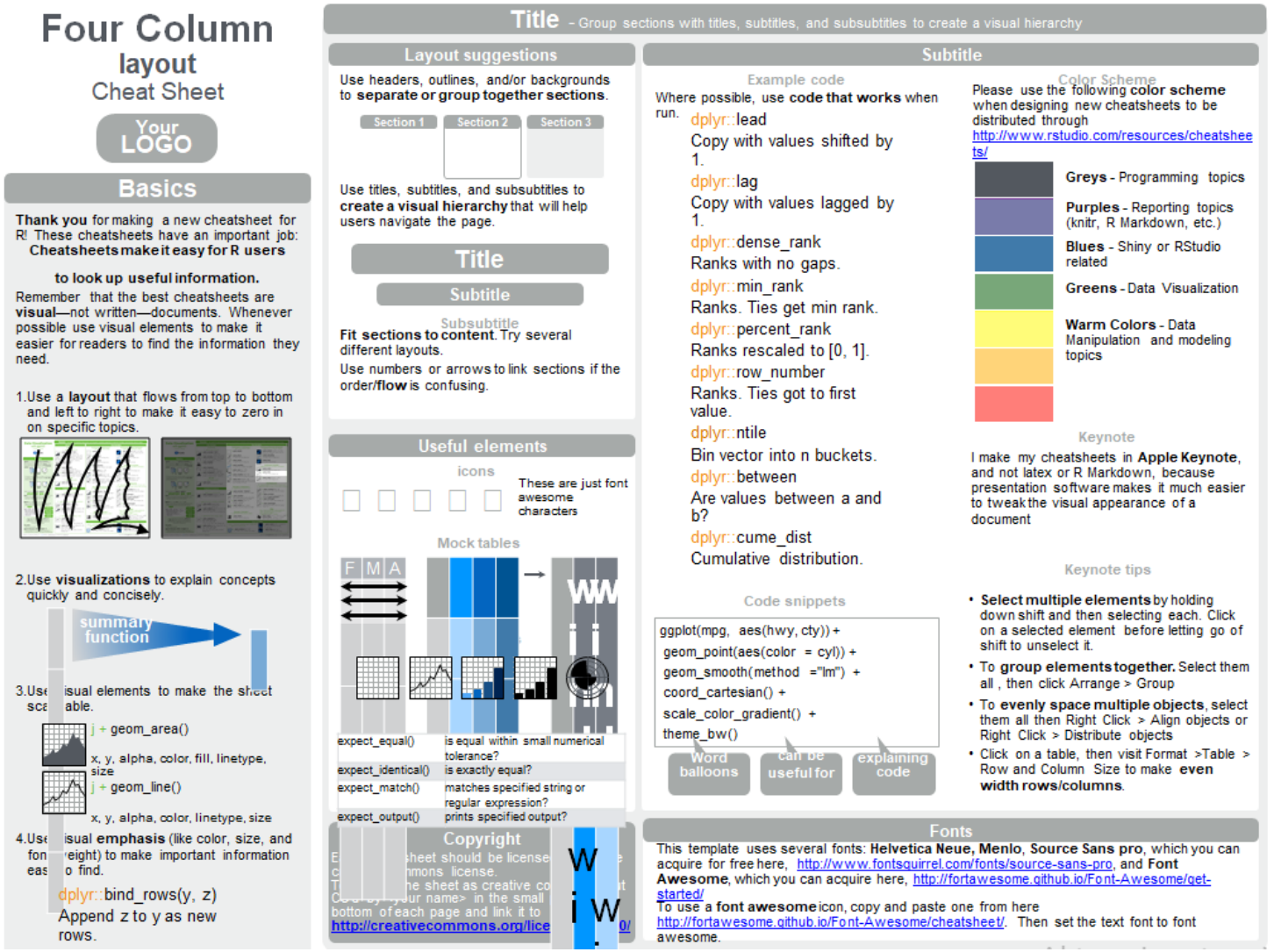


DOWNLOAD APPLE KEYNOTE

DOWNLOAD POWERPOINT

https://www.rstudio.com/resources/cheatsheets/how-to-contribute-a-cheatsheet/

Stary szablon



https://github.com/rstudio/cheatsheets/blob/master/old/power-point-exports/0-template.pptx

Składowe R-owej ściągawki

Four Column Layout : : CHEAT SHEET

Basics

Thank you for making a new cheatsheet for R! These cheatsheets have an important job:

Cheatsheets make it easy for R users to look up useful information.

Remember that the best cheatsheets are **visual**—not written—documents. Whenever possible use visual elements to make it easier for readers to find the information they need.

1. Use a **layout** that flows and makes it easy to zero in on specific topics.

2. Use **visualizations** to explain concepts quickly and concisely.

summary function

3. Use visual elements to make the sheet **scannable**.

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

4. Use visual **emphasis** (like color, size, and font weight) to make important information easy to find.

dplyr::**lag()** - Offset elements by 1
dplyr::**lead()** - Offset elements by -1

Layout Suggestions

Use headers, colors, and/or backgrounds to **separate or group together sections**.

Section 1 | Section 2 | Section 3

Create a visual hierarchy. Help users navigate the page with titles, subtitles, and subsubtitles.

Title

SUBTITLE

SUBSUBTITLE

Fit sections to content. Try several different layouts.

Use numbers or arrows to link sections if the order/**flow** is confusing.

Quickly identify content with a **package hexsticker** (if available)

Copyright

Useful Elements

CODE

Where possible, use **code that works** when run.

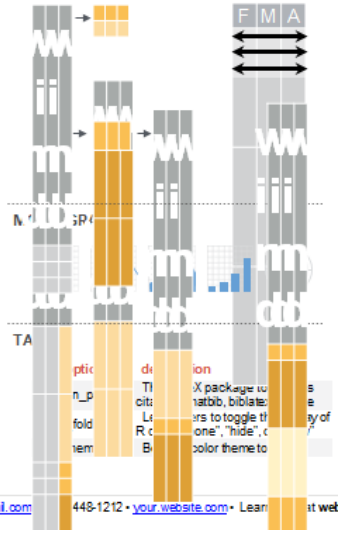
```
ggplot(mpg, aes(hwy, cty)) +
 geom_point(aes(size = fl)) +
 geom_smooth(method ="lm")
```

Word balloons | can help explain | code

ICONS

These are just **font awesome** characters

MOCK TABLES



Logistics

FONTS

This template uses several fonts: **Helvetica Neue, Menlo**, **Source Sans pro**, which you can acquire for free here, www.fontsquirrel.com/fonts/source-sans-pro, and **Font Awesome**, which you can acquire here, fortawesome.github.io/Font-Awesome/get-started/

To use a **font awesome** icon, copy and paste one from here fortawesome.github.io/Font-Awesome/cheatsheet/. Then set the text font to font awesome.

KEYNOTE

I make my cheatsheets in **Apple Keynote**, and not latex or R Markdown, because presentation software makes it much easier to tweak the visual appearance of a document

KEYNOTE TIPS

- **Select multiple elements** by holding down shift and then selecting each. Click on a selected element before letting go of shift to unselect it.
- To **group elements together.** Select them all , then click Arrange > Group
- To **evenly space multiple objects**, select them all then Right Click > Align objects or Right Click > Distribute objects
- Click on a table, then visit Format >Table > Row and Column Size to make **even width rows/columns**.

YOUR LOGO (optional)

https://github.com/rstudio/cheatsheets/blob/master/powerpoints/0-template.pptx

Praca w grupach

https://github.com/kapelner/ICEbox

https://github.com/AppliedDataSciencePartners/xgboostExplainer

https://github.com/cran/nlstools