reportsrender

Gregor Sturm

Dec 02, 2019

CONTENTS:

CHAPTER

ONE

GENERATE REPRODUCIBLE REPORTS FROM RMARKDOWN OR JUPYTER NOTEBOOKS

Reportsrender allows to create reproducible, consistently looking HTML reports from both jupyter notebooks and Rmarkdown files. It makes use of papermill and Rmarkdown to execute notebooks and uses Pandoc to convert them to HTML.

Features:

- two execution engines: papermill and Rmarkdown.
- support any format supported by jupytext.
- create self-contained HTML that can be shared easily.
- hide inputs and/or outputs of cells.
- parametrized reports

See the documentation for more details!

CHAPTER

TWO

GETTING STARTED

- Execute an rmarkdown document to HTML using the Rmarkdown engine

```
reportsrender --engine=rmd my_notebook.Rmd report.html
```

- Execute a parametrized jupyter notebook with papermill

```
reportsrender --engine=papermill jupyter_notebook.ipynb report.html --params="data_
↪file=table.tsv"
```

TODO add example notebooks.

CHAPTER

THREE

USAGE FROM COMMAND LINE

```
reportsrender

Execute and render a jupyter/Rmarkdown notebook.
The `index` subcommand generates an index html
or markdown file that links to html documents.

Usage:
  reportsrender <notebook> <out_file> [--cpus=<cpus>] [--params=<params>] [--engine=
↪<engine>]
  reportsrender index [--index=<index_file>] [--title=<title>] [--] <html_files>...
  reportsrender --help

Arguments and options:
  <notebook>            Input notebook to be executed. Can be any format supported by␣
↪jupytext.
  <out_file>            Output HTML file.
  -h --help             Show this screen.
  --cpus=<cpus>         Number of CPUs to use for Numba/Numpy/OpenBLAS/MKL [default:␣
↪1]
  --params=<params>     space-separated list of key-value pairs that will be passed
                        to papermill/Rmarkdown.
                        E.g. "input_file=dir/foo.txt output_file=dir2/bar.html"
  --engine=<engine>     Engine to execute the notebook. [default: auto]

Arguments and options of the `index` subcommand:
  <html_files>          List of HTML files that will be included in the index. The␣
↪tool
                        will generate relative links from the index file to these␣
↪files.
  --index=<index_file>  Path to the index file that will be generated. Will be
                        overwritten if exists. Will auto-detect markdown (.md) and
                        HTML (.html) format based on the extension. [default: index.
↪html]
  --title=<title>       Headline of the index. [default: Index]

Possible engines are:
  auto                  Use `rmd` engine for `*.Rmd` files, papermill otherwise.
  rmd                   Use `rmarkdown` to execute the notebook. Supports R and
                        python (through reticulate)
  papermill             Use `papermill` to execute the notebook. Works for every
                        kernel available in the jupyter installation.
```

CHAPTER

FOUR

INSTALLATION

4.1 Conda (recommended):

As this package dependes on both R and Python packages, I recommend to install the package through conda.

I yet need to create a conda package and upload it on conda-forge, but you can create the following environment and install the package:

```
conda create -c bioconda -c conda-forge -n reportsrender \
      "python>=3.6" \
      "r-base>=3.5" \
      r-rmarkdown \
      r-reticulate \
      r-bookdown \
      flit \
      pandoc

conda activate reportsrender
flit installfrom github:grst/reportsrender
```

4.2 Manual install:

4.2.1 Get dependencies:

- Python
- pandoc

For the Rmarkdown render engine additionally:

- R and the following packages:

```
rmarkdown
reticulate
```

4.2.2 Install from github:

```
pip install flit
flit installfrom github:grst/reportsrender
```

CHAPTER

FIVE

FEATURES

5.1 Execution engines

5.2 Supported notebook formats

5.3 Hiding cell inputs/outputs

You can hide inputs and or outputs of individual cells:

5.3.1 Papermill engine:

Within a jupyter notebook:

- edit cell metadata
- add one of the following *tags*: *hide_input*, *hide_output*, *remove_cell*

```
{
    "tags": [
        "remove_cell"
    ]
}
```

5.3.2 Rmarkdown engine:

- all native input control options (e.g. *results='hide'*, *include=FALSE*, *echo=FALSE*) are supported. See the Rmarkdown documentation for more details.

Jupytext automatically converts the tags to Rmarkdown options for all supported formats.

5.4 Parametrized notebooks

5.4.1 Papermill engine:

- See the Papermill documentation

Example:

- Add the tag *parameters* to the metadata of a cell in a jupyter notebook.
- Declare default parameters in that cell:

```
input_file = '/path/to/default_file.csv'
```

- Use the variable as any other:

```
import pandas as pd
pd.read_csv(input_file)
```

5.4.2 Rmarkdown engine:

- See the documentation.

Example:

- Declare the parameter to the yaml frontmatter.
- You can set default parameters that will be used when the notebook is executed interactively in Rstudio. They will be overwritten when running through *reportsrender*.

```
---
title: My Document
output: html_document
params:
  input_file: '/path/to/default_file.csv'
---
```

- Access the parameters from the code:

```
read_csv(params$input_file)
```

5.4.3 Be compatible with both engines:

Yes it's possible! You can execute the same notebook with both engines. Adding parameters is a bit more cumbersome though.

Example (Python notebook stored as *.Rmd* file using *jupytext*):

````
---
title: My Document
output: html_document
params:
  input_file: '/path/to/default_file.csv'
---

```{python tags=c("parameters")}
try:
    # try to get param from Rmarkdown using reticulate.
    input_file = r.params["input_file"]
except:
    # won't work if running papermill. Re-declare default parameters.
    input_file = "/path/to/default_file.csv"
```
````

5.5 Sharing reports

github pages. . .

5.6 Combine notebooks into a pipeline

CHAPTER

SIX

USAGE AS PYTHON LIBRARY

Reportsrender provides a public API that can be used to execute and convert notebooks to HTML:

Execute and Render notebooks as HTML reports.

render_rmd(input_file, output_file[, params])	Wrapper function to render an Rmarkdown document with the R *rmarkdown* package and convert it to HTML using pandoc and a custom template.
render_papermill(input_file, output_file[, ...])	Wrapper function to render a jupytext/jupyter notebook with papermill and pandoc.
run_pandoc(in_file, out_file[, res_path, ...])	Convert to HTML using pandoc.

6.1 reportsrender.render_rmd

reportsrender.**render_rmd**(*input_file*, *output_file*, *params=None*)

Wrapper function to render an Rmarkdown document with the R *rmarkdown* package and convert it to HTML using pandoc and a custom template.

Parameters

- **input_file** (str) – path to input (Rmd) file
- **output_file** (str) – path to output (html) file
- **params** (Optional[dict]) – Dictionary that will be passed to *params* arg of *rmarkdown::render*. See https://bookdown.org/yihui/rmarkdown/parameterized-reports.html for more details.

6.2 reportsrender.render_papermill

reportsrender.**render_papermill**(*input_file*, *output_file*, *params=None*)

Wrapper function to render a jupytext/jupyter notebook with papermill and pandoc.

Parameters

- **input_file** (str) – path to input file. Can be any format supported by jupytext.
- **output_file** (str) – path to output (html) file.
- **params** (Optional[dict]) – parameter dictionary that will be passed to papermill. See https://papermill.readthedocs.io/en/latest/usage-parameterize.html for more details.

6.3 reportsrender.run_pandoc

reportsrender.**run_pandoc**(*in_file*, *out_file*, *res_path=None*, *template_file=None*, *css_file=None*)

Convert to HTML using pandoc.

Will create a standalone, self-contained html based on the specified template.

Parameters

- **in_file** (str) – path to input file. Can be any format supported by pandoc. The format will be inferred from the file extension.
- **out_file** (str) – path to output (html) file.
- **res_path** (Optional[Collection[str]]) – List of pandoc resource paths (pandoc will look here for asset files). If no template_file is provided the resource path of the default template will be appended.
- **template_file** (Optional[str]) – path to the pandoc template. Per default, the *adaptive-bootstrap* template shipped with this package will be used.
- **css_file** (Optional[str]) – path to the css file used by pandoc. Per default, the css file from the *adaptive-bootstrap* template shipped with this package will be used.

PYTHON MODULE INDEX

r

INDEX

R