jupyter

June 11, 2020

Sumário

1 Introdução

O *Jupyter Notebook* é a maneira que optei para escrever os códigos na linguagem *Python*, visto que além de rodar os códigos, é possível: 1. Documentar os *scripts*, escrevendo o singnificado e objetivo de cada conjunto de comandos; 2. Atualizar os meus repositórios na plataforma **GitHub**; 3. Trabalhar com uma diversidade de opções de exportação do arquivo em formatos diversos, adaptados até mesmo para as simples leitura, como PDFs e Markdowns.

É no procesos de exportação dos arquivos que eu me ative nessa publicação, pois um dos objetivos de longo prazo que busco é exportar relatórios padronizados, para distribuição geral e irrestrita, ou seja, quero algo que não seja inteligível apenas por pessoas que conhecem de programação.

Para isso foram aqui apresentados um diversidade de opções para exportação de um arquivo *.ipynb*, sendo possível: - Incluir apenas campos determinados; - Incluir apenas as células que tenham determinada *tag*; - Incluir apenas as células de *markdown*; - Excluir as células de *outputs*.

2 Get *Jupyter Notebook* filename

Testei diversos comandos para obter o nome do *Jupyter Notebook* em uma variável. A melhor opção que encontrei estava nesse *post* que tem diversas outras opções.

3 Funções

3.1 Variáveis em *Markdowns*

Para inserir uma variável em uma célula markdow para eu inserir a variável entre colchetes duplos, por exemplo 10. Logo, se eu alterar o valor de a para qualquer um terei que **a=10**.

O mesmo pode ser feito com tabelas. Em tentativa de inserir tabelas diretamente do Pandas não obtive sucesso... Depois temos dataframe modificado pelo *.to_html()*, função que fornece várias opções a serem exploradas.

A

```
<th>B</th>    </tr>  </thead>  <tbody>    <tr>      <td>1.0000</td>      <td>4.12134</td>
```

3.2 Linhas de Tabelas

Descobri que nesse *post* que é possível trabalhar para inserir também mais de uma tabela alinhada.

3.3 Comandos do Sistema

Praticamente qualquer comando do sistema pode ser acessado usando previamente **!**, o qual passa qualquer comando subsequente diretamente para o sistema operacional. Você pode até usar variáveis python em comandos enviados para o sistema operacional!

3.4 HTML

4 *Export*

Os arquivos Jupyter Notebook podem ser exportados em diversos formatos, seja através do menu de opções, ou através dos comandos. Ao exportar, é possível definir diversas opções que limitam o que será exportado, podendo escolher determinados tipos de células ou, até mesmo, células invidivuais.

No *post* Jupyter Notebook nbconvert without Magic Commands/ w/o Markdown é apresentado algumas opções de exportação. Incorporei várias delas no script *../codes/files/export_jupyter.py*. Ainda existem outras opções que não estudei a finalidade, listadas a seguir:

1. *–stdout*
2. *–TemplateExporter.exclude_input_prompt=True*
3. *–TagRemovePreprocessor.remove_input_tags = {"hide"}*

–to pdf: markdown

Usando pandoc descobri que dá pra exportar para **.doc**! Não ficou tão bom, mas ajuda!

5 GitHub

A partir do *post* **How to Git Jupyter Notebooks the Right Way**, compreendi que é considerada como *best pratices* no git de projetos escritos em *Jupyter Notebook* a aplicação de um determinado código usando o package *nbstripout*, conforme apresentado abaixo. No vídeo **nbstripout: strip output from Jupyter and IPython notebooks** é explicado detalhadamente como o comando atua.

Criei uma função para exportar o *Jupyter Notebook* em diversos formatos. Aproveitei para incorporar o comando do `nbstripout` na função que faz o *commit*, visando simplificar as coisas.

6 *Requirements*

O comando `pip freeze` é o mais difundido na internet para se obter os *requirements.txt*, ou seja, o arquivo com o qual é possível indicar quais os *packages* necessários para rodar um determinado *script*.

Tentei usar também o package `pipreqs`, porém ele não funciona em *Juptyter Notebook*. Descobri ainda que o comando `conda env export > environment.yml` pode auxiliar na criação destes parâmetros.

7 Erros

Em uma tentativa de exportar o *Jupyter Notebook* para PDF tive problemas. O arquivo não era exportado e apresentava a seguinte mensagem de erro: - *nbconvert failed: xelatex not found on PATH, if you have not installed xelatex you may need to do so. Find further instructions at https://nbconvert.readthedocs.io/en/latest/install.html#installing-tex.*

Para solucionar, descobri que é necessário instalar, no Linux, akguns pacotes de aplicativos com os seguintes comandos, sendo o primeiro uma instalação mais compacta e o segundo uma instalação completa.

```
sudo apt-get install texlive-xetex texlive-fonts-recommended
texlive-generic-recommended
```

```
sudo apt-get install texlive-full
```

8 Referêcias

Há muita informação na internet sobre funcionalidades do *Jupyter Notebook*. Apenas para exemplificar, usei particialmente algumas das funções e truques apresentados em **Jupyter Notebook Extensions** e **28 Jupyter Notebook Tips, Tricks, and Shortcuts**.