Enrich your Jupyter Notebook with these tips

towardsdatascience.com/enrich-your-jupyter-notebook-with-these-tips-55c8ead25255

November 29, 2021

Practical tips to enhance your workflow documentation

One beautiful feature of Jupyter Notebook (Notebook from here onwards) is the ability to use markdown cells alongside code cells. These markdown cells enable us to document more expressively and clearly so that it's easier for future users to understand the workflow of the Notebook. In this post, I share a few of my tips to enrich documentation in the markdown cells without having to install any extensions.

📝 0. Refresher

If you have been using Notebook, you probably already know the basics of Markdown. In case you need a refresher, here's a one-minute introduction to Markdown on its commonly used syntax:

```
#### Headers# Header 1 ## Header 2#### Styles*Italic*, **bold**, _underscore_,
~~strikethrough~~#### Hyperlink[hyperlink](https://www.markdownguide.org/)#### Table|
Default | Left-aligned | Center-aligned | Right-aligned ||---------|:-------------|:-
-------------:|--------------:|| Default | Left | Center | Right |#### Others*
Bulleted item 1    * Bulleted subitem 1* Bulleted item 2***1. Ordered item 1
1.1. Ordered subitem 1  2. Ordered item 2***- [ ] Unchecked box- [x] Checked box
```

Its rendered output:

Having refreshed the basics, it's time to look at more sophisticated features of Markdown cells to enrich documentation for improved readability.

🎨 1. Colour code text

Big chunk of text in black and white can be discouraging to read. One way to enrich black and white text and improve readability of the text is to add colours to highlight and pop key parts. Here are three different ways to add colour to the text:

🎨 1.1. Colour text

We can change the colour of the text using the html `<font>` tag. We can use colour names or hexadecimal colour code:

```
Example: <font color=green>green text</font>, <font color=blue>*blue italised text*
</font> and <font color=#FF0000>**red bold text**</font>.
```

Example: green text, *blue italised text* and **red bold text**.

If you would like to explore more colour names, this may come in handy. If colour names don't quite capture what you are after, you can explore hexadecimal colours to access a wider range of options. Here's my favourite resource to explore hexadecimal colours.

🎨 1.2. Highlight text

We can also highlight text with the html `<mark>` tag:

```
In addition, we can also <mark>highlight text</mark>.
```

It's now easier to draw attention to the highlighted part of the text.

In addition, we can also highlight text.

🎨 1.3. Use alerts

Lastly, we can format the background and font colour using bootstrap alert to make the text documentation more engaging to read:

```
<div class="alert alert-info">Example text highlighted in blue background.</div><div
class="alert alert-success">Example text highlighted in green background.</div><div
class="alert alert-warning">Example text highlighted in yellow background.</div><div
class="alert alert-danger">Example text highlighted in red background.</div>
```

Example text highlighted in blue background.

Example text highlighted in green background.

Example text highlighted in yellow background.

Example text highlighted in red background.

These formats are beautiful! Adding colours can immediately improve the readability of your Notebook documentation by making key points stand out and making the documentation less boring. This way, it becomes easier to skim and get the main points quickly.

📍 2. Format text appropriately

Another way to enrich documentation is to use suitable richer formats of texts. Let's look at three different ways to format text:

📍 2.1 Insert math equation with LaTeX

There is often a need to reference mathematical equations in Notebook documentations. With `$`, we can use LaTeX to show nicely formatted math formulas:

```
$$logloss(\theta) = - {1 \over m} \sum_{i=1}^m (y_i  \ln(\hat p(y_i=1)) + (1-y_i)
\ln(1-\hat p(y_i=1)))$$
```

$$logloss(\theta) = -\frac{1}{m}\sum_{i=1}^{m}(y_i \ln(\hat{p}(y_i = 1)) + (1 - y_i) \ln(1 - \hat{p}(y_i = 1)))$$

When the equation is wrapped with double `$`, it will be centre-aligned. If we use single `$`, it will be left-aligned. Alternatively, we can also use this syntax to write the formula:

```
\begin{equation} logloss(\theta) = - {1 \over m} \sum_{i=1}^m (y_i  \ln(\hat
p(y_i=1)) + (1-y_i)  \ln(1-\hat p(y_i=1))) \end{equation}
```

If you are not familiar with LaTeX, check out this guide or this one to get started.

📍 2.2. Use code-blocks

At times, it's useful to show code references in markdown cells instead of running them in code cells. We can use single backticks `` ` `` to show a code block inline:

```
If you haven't installed textblob, you can install it with: `pip install texblob`.
```

If you haven't installed textblob, you can install it with: `pip install texblob`.

For a bigger chunk of code, we can use triple backticks ```` ``` ````:

```
If you haven't installed libraries, you can install them with the following
command:```conda install pandas, numpy, sklearn -ypip install textblob```
```

If you haven't installed libraries, you can install them with the following command:

```
conda install pandas, numpy, sklearn -y
pip install textblob
```

If we specify the language name after the first triple `` ` ``, code block gets formatted colourfully where applicable:

```
```python{"minimum": 10, "maximum": 50, "name": "optimiser"}```
```
```

```
{"minimum": 10, "maximum": 50, "name": "optimiser"}
```

📍 2.3. Use quote indentation

Indentation is another way to format text to improve readability. We can add indentation with `>`:

```
Sample non-indented sentence here.> **Note:** Indented text.
```

Sample non-indented sentence here.

> **Note:** Indented text.

📹 3. Add media

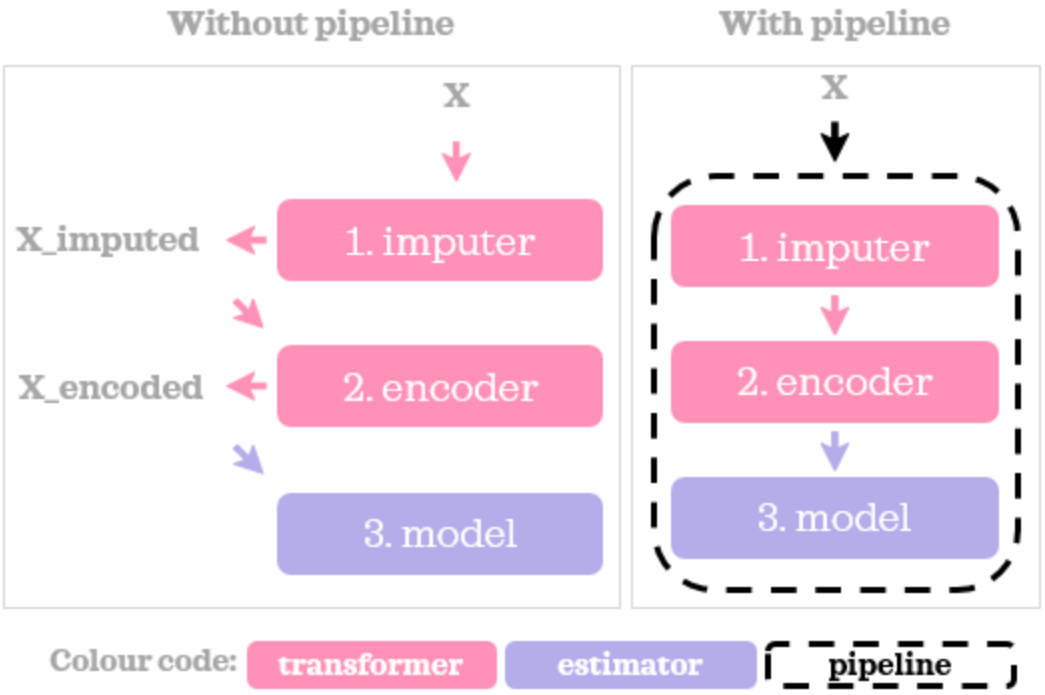
> 'A picture is worth a thousand words.'

Documentation doesn't always have to be in words. Images and other media can help us communicate ideas that are otherwise difficult to express with text. Adding relevant media is another great way to enrich documentation with the necessary information for future users.

📷 3.1. Embed images including GIFs

We can add images to markdown cells using the html `<img>` tag:

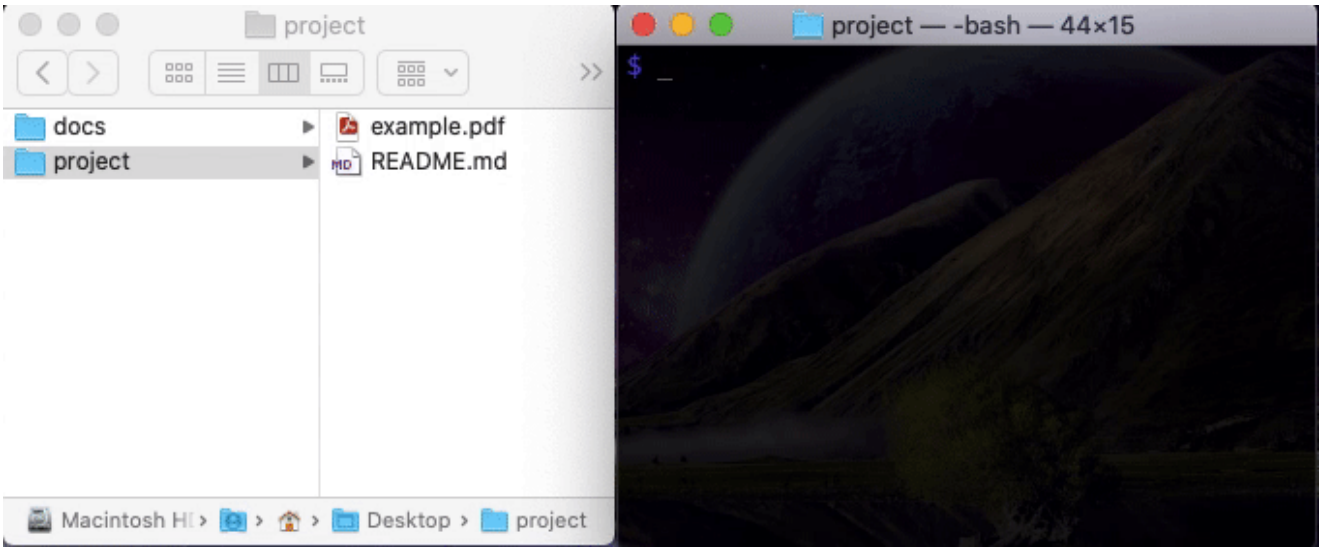
```
<img src="example.jpeg">
```



Here, `example.jpeg` is saved in the same folder as the notebook. We can change the size of the image use `width` or `height` arguments. For instance, `<img src="example.jpeg" width=500>` will resize the image displayed to the desired width while maintaining the width to height ratio.

If adding diagrams, charts or other data visualisations in static image doesn't quite capture what we want to convey, we can embed GIF, an animated image using the exact same syntax structure:

```
<img src="example.gif"/>
```



The path of the file can be a web link too:

```
<img src="">
```

📹 3.2. Embed videos

> ‘A (one-minute) video is worth 1.8 million words.’

If GIFs are not enough, the next level is to use videos. We can use the html `<video>` tag to display a video:

```
<video controls src="example.mp4"  width=600/>
```

For instance, screen recording how to complete a task, saving it as video file and embedding the video in the Notebook can be useful for future users.

⚪ 3.3. Add shapes and emojis

A long paragraph of plain text can be boring and not appealing to read. Adding shapes and emojis tastefully can make text more interesting and appealing to read:

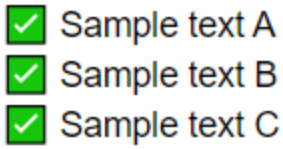
```
&#10148; Bullet point one</br>&#10148; Bullet point two</br>&#10148; Bullet point three
```

```
✔️ Sample text A</br>✔️ Sample text B</br>✔️ Sample text C
```

Check out this to explore more shapes (and emojis). This emoji cheat-sheet is useful when searching emoji by name.

➤ Bullet point one
➤ Bullet point two
➤ Bullet point three

Voila! These were my tips for enriching Jupyter Notebook documentation.✨ We may not use all of these features at once. But knowing how to use these prepares you when the right opportunity arises.



Thank you for reading my post. If you want to learn more about Markdown, check out this guide. If you are interested, here are links to some of my posts:

- Organise your Jupyter Notebook with these tips
- Useful IPython magic commands
- Introduction to Python Virtual Environment for Data Science
- Introduction to Git for Data Science
- Simple data visualisations in Python that you will find useful
- 6 simple tips for prettier and customised plots in Seaborn (Python)
- 5 tips for pandas users
- Writing 5 common SQL queries in pandas

Bye for now 🏃💨