# A Large-Scale Study About Quality and Reproducibility of Jupyter Notebooks

| | |
|---|---|
| **Typ** | Konferensartikel |
| **Författare** | Joao Felipe Pimentel |
| **Författare** | Leonardo Murta |
| **Författare** | Vanessa Braganholo |
| **Författare** | Juliana Freire |
| **Webbadress** | https://ieeexplore.ieee.org/document/8816763/ |
| **Ort** | Montreal, QC, Canada |
| **Utgivare** | IEEE |
| **Sidor** | 507-517 |
| **ISBN** | 978-1-72813-412-3 |
| **Datum** | 5/2019 |
| **DOI** | 10.1109/MSR.2019.00077 |
| **Hämtad den** | 2019-09-30 13:12:03 |
| **Bibliotekskatalog** | Crossref |
| **Namn på konferens** | 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR) |
| **Språk** | en |
| **Sammanfattning** | Jupyter Notebooks have been widely adopted by many different communities, both in science and industry. They support the creation of literate programming documents that combine code, text, and execution results with visualizations and all sorts of rich media. The self-documenting aspects and the ability to reproduce results have been touted as significant benefits of notebooks. At the same time, there has been growing criticism that the way notebooks are being used leads to unexpected behavior, encourage poor coding practices, and that their results can be hard to reproduce. To understand good and bad practices used in the development of real notebooks, we studied 1.4 million notebooks from GitHub. We present a detailed analysis of their characteristics that impact reproducibility. We also propose a set of best practices that can improve the rate of reproducibility and discuss open challenges that require further research and development. |
| **Protokolltitel** | 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR) |
| **Tillagd den** | 2019-09-30 13:12:03 |
| **Ändrad den** | 2019-10-07 12:52:22 |

## Etiketter:

GitHib mining, Jupyter

### Bifogade dokument

- Pimentel m. fl. - 2019 - A Large-Scale Study About Quality and Reproducibil.pdf

  Tidigare kritik mot Jupyter notebooks inkluderar:

  - "hidden states"
  - oväntad exekveringsordning med fragmenterad kod
  - dåliga practices för namngivning, versionshantering, testning och modularisering av kod
  - beroenden (med rätt version) är inte inkodade i formatet

  Författarna utgår från detta och undersöker Jupyter notebooks på GitHub med målet att se hur reproducerbara Jupyter notebooks är.

  De utgår från de notebooks som lades upp på GitHub mellan 1/1 2013 (året då Jupyter notebooks lanserades) och 16/4 2018, 1 450 071 stycken. De exkluderar ogiltiga och tomma notebooks, samt duplicerade dito. De sistnämnda hittar de genom att jämföra hashar.

  De hävdar att 43 204 notebooks inte specar språk. 330 378 av dessa har nbformat<4 som är den första språkagnostiska Jupyterreleasen. Det senare borde innebära att de använder Python, men författarna exkluderar dessa ur datasetet.

  De kommer fram till att:

- fördelningen av antal notebooks/repro är väldigt högerskeva
- de mest förekommande språken är Python (93,32%), R (1,31%) och Julia (0,93%).
- de flesta notebooksen innehåller markdownceller och knappt 1/4 av alla celler är markdowndito. Texten i dessa är dock ofta kort och innehåller inte features som listor, länkar och bilder. De konstaterar att innehållet i markdowncellerna kanske inte räcker för välbeskrivna narrativ.
- notebooksens namn är ofta korta men meningsfulla, men innehåller ofta olämpliga tecken, d.v.s. annat än [A-Z a-z 0-9 ._-].
- de mest frekvent importerade modulerna i Pythonnotebooksen är numpy, matplotlib och pandas.
- 53,94% respektive 8,54% av alla syntaktiskt korrekta Pythonnotebooks definierar funktioner respektive klasser. Dock är det vanligare med funktioner i notebooks som även innnehller loopar eller villkorssatser (d.v.s. ett lite mer komplicerat programflöde). Funktionerna extraheras sällan i lokala moduler.
- enligt en ganska grov uppskattning importerar endast 1,54% av de syntaktiskt korrekta Pythonnotebooksen testmoduler. (Det oftast importerade testpaketet är en modul från en kurs, som har forkats 3211 gånger. :-D) Författarna förklarar detta med att vetenskaplig kod ofta är svårtestad. (Jag är inte övertygad om att det är en giltig ursäkt!) Men de tycker att en testsvit är viktig för reproducerbarhetens skull.
- de flesta notebooks innehåller output, vilket författarna tycker är bra för reproducerbarheten eftersom det gör att andra kan jämföra sina resultat med dessa.
- bland de notebooks som hade exekverats (d.v.s. innehöll info om i vilken ordning cellerna har exekverats) hade 21,11% icke-exekverade celler och 62,08% hade tomma celler. Antalet icke-exekverade och tomma celler ökar mot slutet av notebooksen.
- i 36,36% av de notebooks som hade en otvetydig exekveringsordning hade cellerna exekverats i en annan ordning än uppifrån och ner.

De försöker reproducera resultaten i 788 718 notebooks, vilket är antalet som inte är korrupta och inte hade trasiga beroenden. De avbröt exekveringen om den tog mer än 5 miunter, vilket var fallet för 9982 stycken. 570 476 exekveringar kastade undantag. De vanligaste undantagen var, i ordning, ImportError, NameError (d.v.s. odefinierad variabel), ModuleNotFoundError och FileNotFoundError. De odefinierade variablerna har förmodligen att göra med exekveringsordningen och "hidden states".

Sammantaget exekverade de 208 323 av notebooksen. Endast 34 836 av dessa gav samma resultat som förlagan, trots att studiens författare exekverade cellerna i samma ordning som de var exekverade av notebookförfattarna.

De föreslår även ett antal "best practices" som ungefär säger att och hur man ska undvika de misstag som beskrivs ovan.

Data och kod finns tillgängligt online!