

Interactive

September 10, 2019

[Data Science Workflow](#) < | > [Hall of Fame](#)

1 Exploring Subsets Interactively

In this notebook you can select criteria for a subset to inspect and compare to the entire dataset.

Instructions:

1. Run Cell 1, wait for "DONE IN <time>" message before continuing.
2. Run Cell 2, fill out criteria before continuing.
3. Run Cell 3, view report.
4. Repeat steps 2 and 3 with new criteria, if desired.

Prerequisites:

- four final CSV file local in ./data_final
 - all aggregations created by aggregate.py local in ./analysis_data
 - ipywidgets
 - pip install ipywidgets
 - nodejs
 - conda install nodejs
 - npm
 - pip install npm
 - labextension
 - for jupyterlab: jupyter labextension install @jupyter-widgets/jupyterlab-manager
 - for jupyter notebook: notebook extension (jupyter nbextension enable --py widgetsnbextension)
-

1.1 Run Cell 1

This takes about 10 minutes. Wait for "DONE IN <time>" message before continuing!

```
[1]: import interactive
import load_data
import datetime
from IPython.core.display import HTML
```

```
# 10 minutes to load data
start = datetime.datetime.now()
data_frames = interactive.data()
end = datetime.datetime.now()
print('\n'+ '-'*80+'\n'+ 'DONE IN {0}'.format(end - start))
```

```
Notebooks loaded in 0:00:33.673711
Repos loaded in 0:00:05.437692
Owners loaded in 0:00:00.777930
Notebook imports loaded in 0:00:43.048986
Errors loaded in 0:00:06.050018
Cell stats loaded in 0:00:04.898696
Cell order loaded in 0:00:35.426024
Outputs loaded in 0:00:06.025272
Statuses loaded in 0:00:03.127332
Cell stats loaded in 0:00:01.842215
Collaboration statuses loaded in 0:00:00.071805
Special functions loaded in 0:00:17.884992
Framework uses loaded in 0:00:11.105059
Educational status loaded in 0:00:00.185618
```

```
-----
DONE IN 0:09:20.538907
```

```
[ ]: # query = interactive.interactive(data_frames)
```

1.2 Run Cell 3

This takes about 1 minute. View report! Repeat Cells 2 and 3 with different criteria.

```
[3]: data_frames_sub = interactive.subset(data_frames, query)
print('\n'+ '-'*73+'\n')
interactive.report_comparisons(data_frames_sub, data_frames)
```

```
Subsetting to Python notebooks pushed between 2019-01-01 and 2019-07-14.
1,515,109 (39.01%) notebooks fit your criteria.
```

1.2.1 Summary Statistics

	num_cells	forks_count	open_issues_count	stargazers_count	\
mean	29.6	10.07	0.92	19.02	
median	19.0	0.00	0.00	0.00	
min	0.0	0.00	0.00	0.00	
max	1641.0	17634.00	2003.00	22831.00	

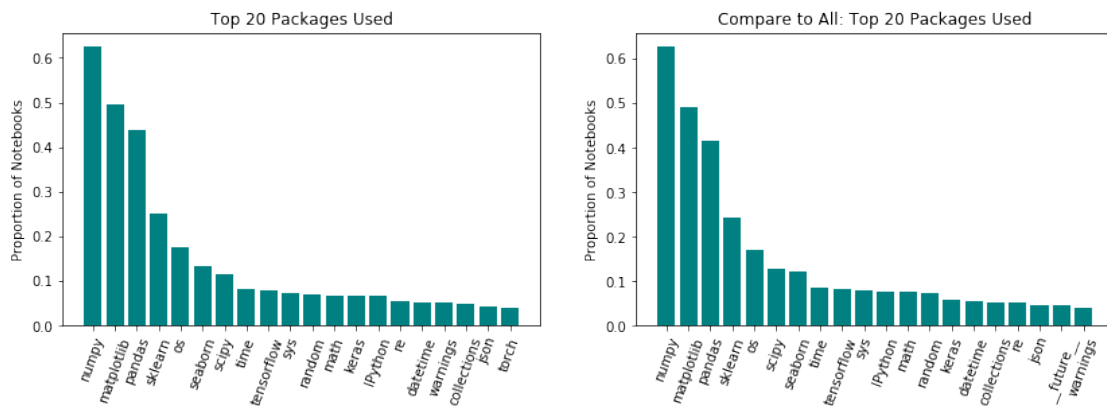
	subscribers_count	watchers_count	lines_of_code	num_words
mean	2.95	19.02	145.55	374.94
median	0.00	0.00	85.00	53.00
min	0.00	0.00	1.00	0.00
max	2446.00	22831.00	72415.00	200404.00

Compare to all:

	num_cells	forks_count	open_issues_count	stargazers_count	\
mean	28.76	5.56	0.49	9.88	
median	19.00	0.00	0.00	0.00	
min	0.00	0.00	0.00	0.00	
max	1641.00	17634.00	2003.00	22831.00	

	subscribers_count	watchers_count	lines_of_code	num_words
mean	2.13	9.88	147.26	405.18
median	1.00	0.00	88.00	60.00
min	0.00	0.00	1.00	0.00
max	2446.00	22831.00	462118.00	200404.00

1.2.2 Package Use

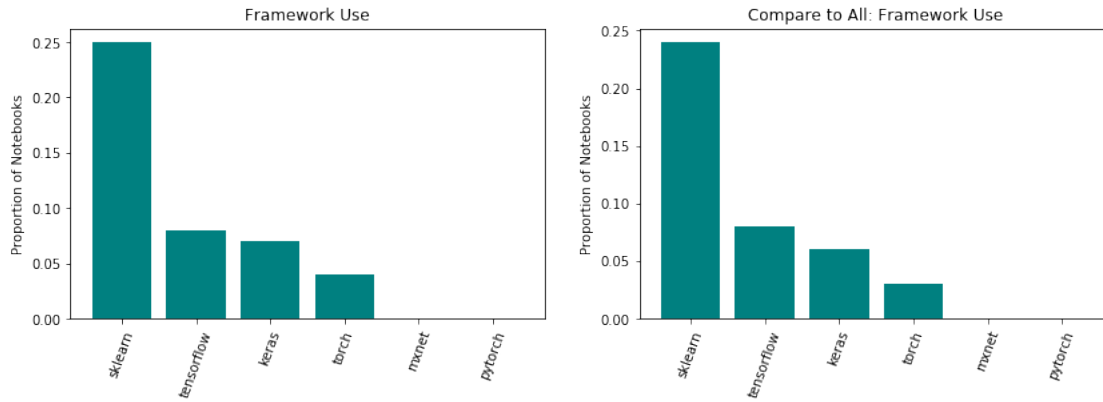


1.2.3 Framework Use

37.39% of these notebooks use at least one framework.

Compare to all:

35.48% of all notebooks use at least one framework.

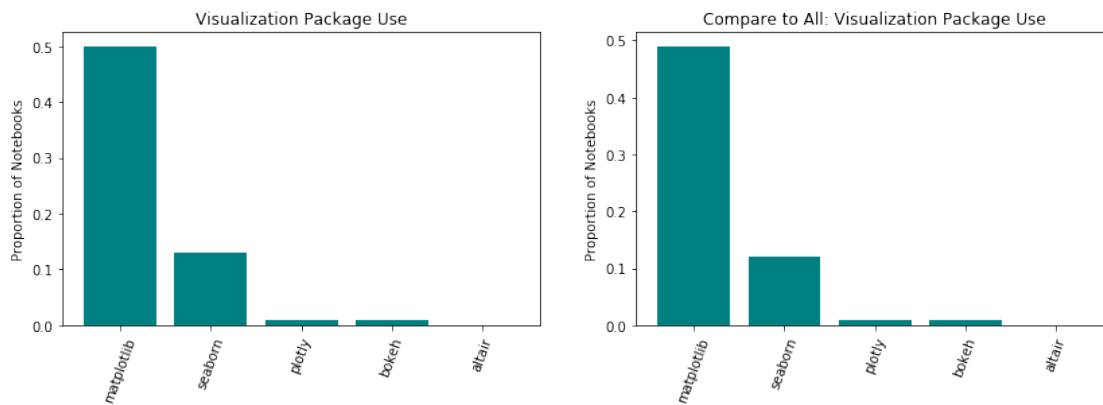


1.2.4 Visualization Package Use

51.09% of these notebooks use at least one visualization package.

Compare to all:

50.71% of all notebooks use at least one visualization package.



1.2.5 Number of Errors per Notebook

```
mean      0.2
median    0.0
min       0.0
max      9104.0
Name: num_errors, dtype: float64
```

Compare to all:

```
mean          0.22
median        0.00
min           0.00
max           9104.00
Name: num_errors, dtype: float64
```

1.2.6 Ratio of Markdown to Code

```
mean          6.26
median        0.66
min           0.00
max           18029.00
Name: ratio_wl, dtype: float64
```

Compare to all:

```
mean          6.45
median        0.69
min           0.00
max           18029.00
Name: ratio_wl, dtype: float64
```

1.2.7 Execution Order

74.71% of these notebooks have cells run in order.

83.96% of these notebooks have at least one output, 70.14% of which are run in order.

85.12% of these notebooks were able to be parsed with Python AST.

Of these, 12.05% had a function used before it was defined, 1.74% had a package used before it was imported, and 4.72% used a variable before it was defined.

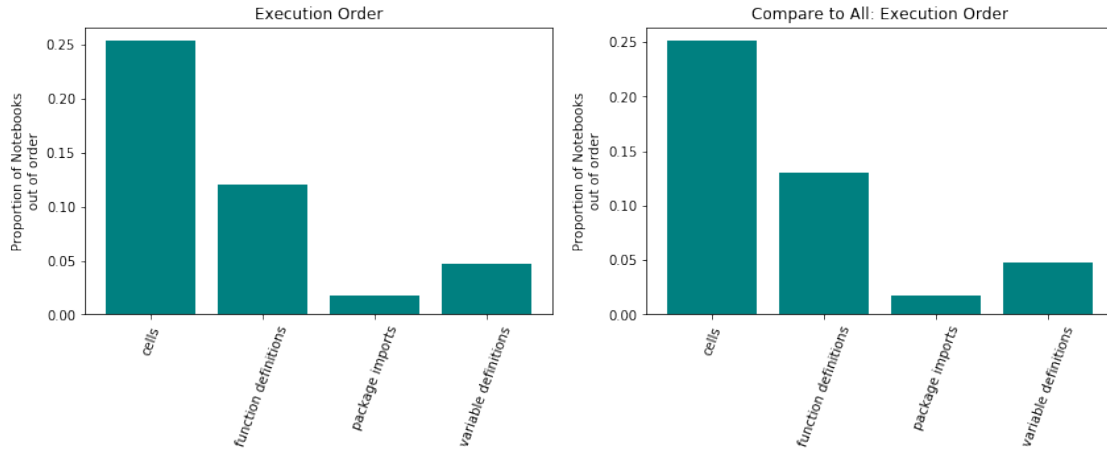
Compare to all:

74.92% of all notebooks have cells run in order.

84.4% of all notebooks have at least one output, 70.58% of which are run in order.

86.41% of all notebooks were able to be parsed with Python AST.

Of these, 13.0% had a function used before it was defined, 1.8% had a package used before it was imported, and 4.76% used a variable before it was defined.

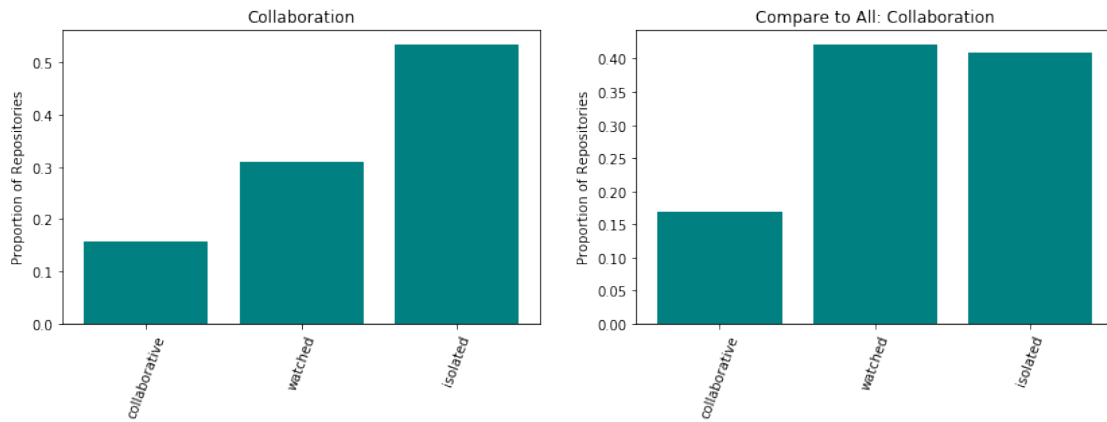


1.2.8 Collaboration

15.63% of these repositories are collaborative, containing 21.19% of these notebooks.

Compare to all:

16.85% of all repositories are collaborative, containing 22.76% of all notebooks.



1.2.9 Educational Status

22.29% of these repos are educational, holding 27.47% of these notebooks

Compare to all:

23.67% of all repos are educational, holding 29.21% of all notebooks

[Data Science Workflow](#) < | > [Hall of Fame](#)