Inference-1

May 16, 2022

[98]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[99]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import os
import glob
```

0.1 Exploratory Part 1 - Flight Cancellations vs Covid Cases

Here we check how weekly flight cancellations and weekly number of covid cases in the first 6 months of 2020 look like and if there is any correlation between them.

[100]:
```
# Let us load flights data for the two states
%cd /content/drive/Shareddrives/CSE544_Project/Exploratory/flight_dataset
df_md_flights_2020 = pd.read_csv("MD_2020_flights.csv", usecols=['FL_DATE',␣
↪'CANCELLED'])
df_la_flights_2020 = pd.read_csv("LA_2020_flights.csv", usecols=['FL_DATE',␣
↪'CANCELLED'])
# Let us load cases data for the two states
%cd /content/drive/Shareddrives/CSE544_Project/covid_dataset
df_covid_la = pd.read_csv('covid_la_cleaned.csv', usecols =␣
↪['submission_date','new_case'])
df_covid_md = pd.read_csv('covid_md_cleaned.csv', usecols =␣
↪['submission_date','new_case'])
```

```
/content/drive/Shareddrives/CSE544_Project/Exploratory/flight_dataset
/content/drive/Shareddrives/CSE544_Project/covid_dataset
```

Let us now process the covid dataset

[101]:
```
#  code to convert time/date sttring to datetime format for easier processing
df_la_flights_2020['FL_DATE'] = pd.to_datetime(df_la_flights_2020['FL_DATE'])
df_md_flights_2020['FL_DATE'] = pd.to_datetime(df_md_flights_2020['FL_DATE'])
df_covid_la['submission_date'] = pd.to_datetime(df_covid_la['submission_date'])
df_covid_md['submission_date'] = pd.to_datetime(df_covid_md['submission_date'])
```

[102]:
```
df_covid_la['day'] = df_covid_la.submission_date.dt.day
df_covid_la['week'] = df_covid_la.submission_date.dt.week
df_covid_la['month'] = df_covid_la.submission_date.dt.month
df_covid_la['year'] = df_covid_la.submission_date.dt.year

df_covid_md['day'] = df_covid_md.submission_date.dt.day
df_covid_md['week'] = df_covid_md.submission_date.dt.week
df_covid_md['month'] = df_covid_md.submission_date.dt.month
df_covid_md['year'] = df_covid_md.submission_date.dt.year
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning:
Series.dt.weekofyear and Series.dt.week have been deprecated.  Please use
Series.dt.isocalendar().week instead.

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: FutureWarning:
Series.dt.weekofyear and Series.dt.week have been deprecated.  Please use
Series.dt.isocalendar().week instead.
  import sys
```

[103]:
```
# Let us see how the data looks like for Maryland's daily covid cases for the␣
 ↪year 2020
df_covid_md = df_covid_md[df_covid_md['year'] == 2020]
# Let us see how the data looks like for Louisiana's daily covid cases for the␣
 ↪year 2020
df_covid_la = df_covid_la[df_covid_la['year'] == 2020]
```

[104]:
```
# We drop the rest of the date time columns to keep only week for our inference.
df_covid_la = df_covid_la[['week','new_case']]
df_covid_md = df_covid_md[['week','new_case']]
```

Let us now process the flights dataset

[105]:
```
df_la_flights_2020['day'] = df_la_flights_2020.FL_DATE.dt.day
df_la_flights_2020['week'] = df_la_flights_2020.FL_DATE.dt.week
df_la_flights_2020['month'] = df_la_flights_2020.FL_DATE.dt.month
df_la_flights_2020['year'] = df_la_flights_2020.FL_DATE.dt.year
df_la_flights_2020 = df_la_flights_2020[['week','CANCELLED']]
df_md_flights_2020['day'] = df_md_flights_2020.FL_DATE.dt.day
df_md_flights_2020['week'] = df_md_flights_2020.FL_DATE.dt.week
df_md_flights_2020['month'] = df_md_flights_2020.FL_DATE.dt.month
df_md_flights_2020['year'] = df_md_flights_2020.FL_DATE.dt.year
```

```
df_md_flights_2020 = df_md_flights_2020[['week','CANCELLED']]
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: FutureWarning:
Series.dt.weekofyear and Series.dt.week have been deprecated.  Please use
Series.dt.isocalendar().week instead.

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: FutureWarning:
Series.dt.weekofyear and Series.dt.week have been deprecated.  Please use
Series.dt.isocalendar().week instead.
  import sys
```

[106]:
```
# Let's sum all the cancelled as well as total scheduled flights by week
df_la_covid_weekly_cases = df_covid_la.groupby(['week']).sum().reset_index()
df_md_covid_weekly_cases = df_covid_md.groupby(['week']).sum().reset_index()
```

[107]:
```
df_la_flights_weekly_cancels = df_la_flights_2020.groupby(['week']).sum().
 ↪reset_index()
df_md_flights_weekly_cancels = df_md_flights_2020.groupby(['week']).sum().
 ↪reset_index()
df_la_flights_weekly_cancels['total'] = df_la_flights_2020.groupby(['week']).
 ↪count().reset_index()['CANCELLED']
df_md_flights_weekly_cancels['total'] = df_md_flights_2020.groupby(['week']).
 ↪count().reset_index()['CANCELLED']
```

We merge the weekly counts for cancelled flights, total flights and covid cases for that week for both of the states

[108]:
```
df_covid_flights_md = pd.merge(df_md_flights_weekly_cancels,
                               df_md_covid_weekly_cases[['week', 'new_case']],
                               on='week',
                               how='left')
df_covid_flights_md['new_case'] = df_covid_flights_md['new_case'].fillna(0)
```

[109]:
```
df_covid_flights_md['cancel/flight'] =  (df_covid_flights_md['CANCELLED'] /␣
 ↪df_covid_flights_md['total']) * 100
df_covid_flights_md['cancel/flight'] = df_covid_flights_md['cancel/flight'].
 ↪apply(lambda x:round(x,2))
df_covid_flights_md
```

[109]:

	week	CANCELLED	total	new_case	cancel/flight
0	1	3	1391	0.0	0.22
1	2	28	1815	0.0	1.54
2	3	20	1791	0.0	1.12
3	4	5	1801	0.0	0.28
4	5	0	1793	0.0	0.00
5	6	20	1805	0.0	1.11
6	7	13	1790	0.0	0.73

7	8	7	1832	0.0	0.38
8	9	15	1835	0.0	0.82
9	10	2	1866	5.0	0.11
10	11	19	1988	26.0	0.96
11	12	253	1994	213.0	12.69
12	13	668	1991	995.0	33.55
13	14	805	1863	2370.0	43.21
14	15	731	1595	4616.0	45.83
15	16	575	1257	4605.0	45.74
16	17	615	1194	5751.0	51.51
17	18	578	1139	6881.0	50.75
18	19	23	711	7125.0	3.23
19	20	6	716	6217.0	0.84
20	21	8	764	7509.0	1.05
21	22	5	772	6465.0	0.65
22	23	5	822	5195.0	0.61
23	24	1	1080	3728.0	0.09
24	25	0	1178	2605.0	0.00
25	26	1	1349	2471.0	0.07
26	27	0	391	2855.0	0.00

[110]:
```
df_covid_flights_la = pd.merge(df_la_flights_weekly_cancels,
                               df_la_covid_weekly_cases[['week', 'new_case']],
                               on='week',
                               how='left')
df_covid_flights_la['new_case'] = df_covid_flights_la['new_case'].fillna(0)
```

[111]:
```
df_covid_flights_la['cancel/flight'] =  (df_covid_flights_la['CANCELLED'] /␣
 ↪df_covid_flights_la['total']) * 100
df_covid_flights_la['cancel/flight'] = df_covid_flights_la['cancel/flight'].
 ↪apply(lambda x:round(x,2))
df_covid_flights_la
```

[111]:

	week	CANCELLED	total	new_case	cancel/flight
0	1	7	1139	0.0	0.61
1	2	33	1573	0.0	2.10
2	3	39	1541	0.0	2.53
3	4	4	1556	0.0	0.26
4	5	10	1539	0.0	0.65
5	6	19	1561	0.0	1.22
6	7	20	1639	0.0	1.22
7	8	19	1758	0.0	1.08
8	9	2	1764	0.0	0.11
9	10	5	1737	0.0	0.29
10	11	6	1702	101.0	0.35
11	12	237	1704	934.0	13.91
12	13	758	1696	2505.0	44.69

13	14	725	1418	9470.0	51.13
14	15	503	1051	7585.0	47.86
15	16	392	842	3333.0	46.56
16	17	364	762	2904.0	47.77
17	18	261	665	2551.0	39.25
18	19	17	447	2290.0	3.80
19	20	23	438	2825.0	5.25
20	21	10	440	2794.0	2.27
21	22	3	440	2624.0	0.68
22	23	25	450	2900.0	5.56
23	24	12	527	3803.0	2.28
24	25	1	534	3159.0	0.19
25	26	0	548	6458.0	0.00
26	27	0	156	8990.0	0.00

Pearson Correlation: In order to understand the coorelation between weekly covid cases and flight cancellations or scheduled flights we perform pearson correlation measure of linear correlation between two sets of data.

[112]:
```
def pearson_correlation(X, Y):
  X_diff = X - X.mean()
  Y_diff = Y - Y.mean()
  XY_diff_sum = (X_diff*Y_diff).sum()
  X_diff_2 = (X_diff ** 2).sum()
  Y_diff_2 = (Y_diff ** 2).sum()
  return XY_diff_sum / np.sqrt((X_diff_2*Y_diff_2))
```

Observation :

[113]:
```
print('Pearson Correlation for Louisiana [Ratio of Cancelled Flights]:
 ↪',pearson_correlation(df_covid_flights_la['cancel/flight'],␣
 ↪df_covid_flights_la['new_case']))
print('Pearson Correlation for Maryland [Ratio of Cancelled Flights]:
 ↪',pearson_correlation(df_covid_flights_md['cancel/flight'],␣
 ↪df_covid_flights_md['new_case']))
```

```
Pearson Correlation for Louisiana [Ratio of Cancelled Flights]:
0.4654271185288054
Pearson Correlation for Maryland [Ratio of Cancelled Flights]:
0.3547660432351594
```

1. We see a change in number of flights cancelled per week per flights scheduled per week as we see an increase in the cases after week 10. We perform a person correlation here and find it to be **positively correlated**.

[114]:

```
print('Pearson Correlation for Louisiana [Total Scheduled Flights]:
 ↪',pearson_correlation(df_covid_flights_la['total'],␣
 ↪df_covid_flights_la['new_case']))
print('Pearson Correlation for Maryland [Total Scheduled Flights]:
 ↪',pearson_correlation(df_covid_flights_md['total'],␣
 ↪df_covid_flights_md['new_case']))
```

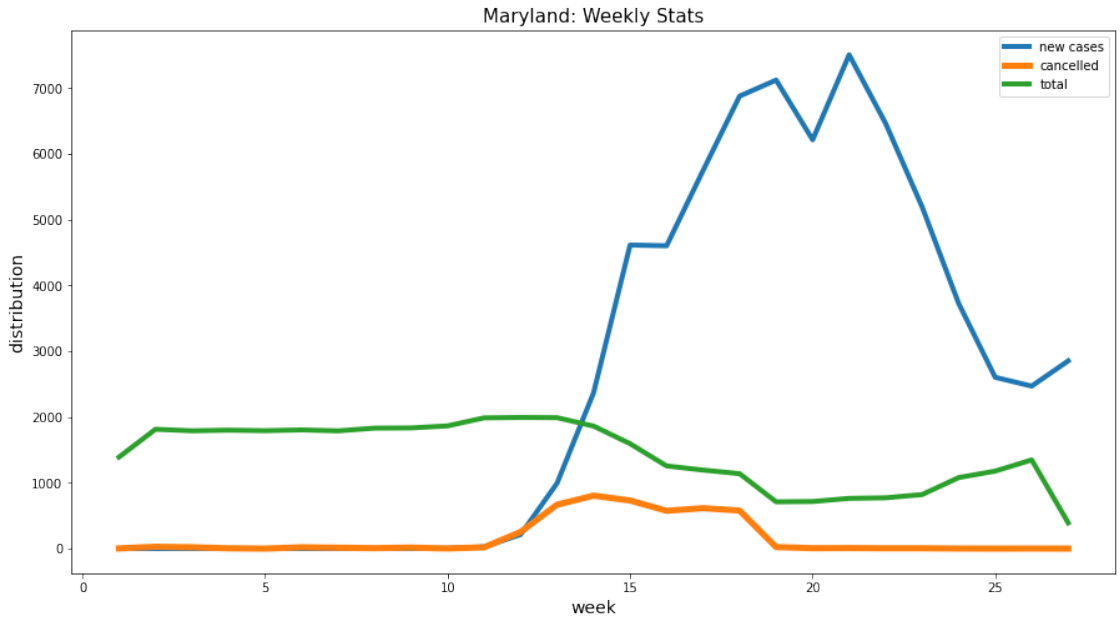
```
Pearson Correlation for Louisiana [Total Scheduled Flights]: -0.5573626240781092
Pearson Correlation for Maryland [Total Scheduled Flights]: -0.8111527340901258
```

2. Secondly, we see a decrease in number of flights per week as we see an increase in the cases after week 10. We perform a person correlation here and find it to be **negatively correlated**.

[115]:
```
fig = plt.figure(figsize=(15,8))
plt.xlabel('week', fontsize=14)
plt.ylabel('distribution', fontsize=14)
x = np.linspace(0, 3000)

plt.plot(df_covid_flights_md['week'], df_covid_flights_md['new_case'],␣
 ↪label="new cases", linewidth = 4)
plt.plot(df_covid_flights_md['week'], df_covid_flights_md['CANCELLED'],␣
 ↪label="cancelled", linewidth = 5)
plt.plot(df_covid_flights_md['week'], df_covid_flights_md['total'],␣
 ↪label="total", linewidth = 4)
plt.title("Maryland: Weekly Stats", fontsize = 15)
plt.legend(fontsize = 10)
```

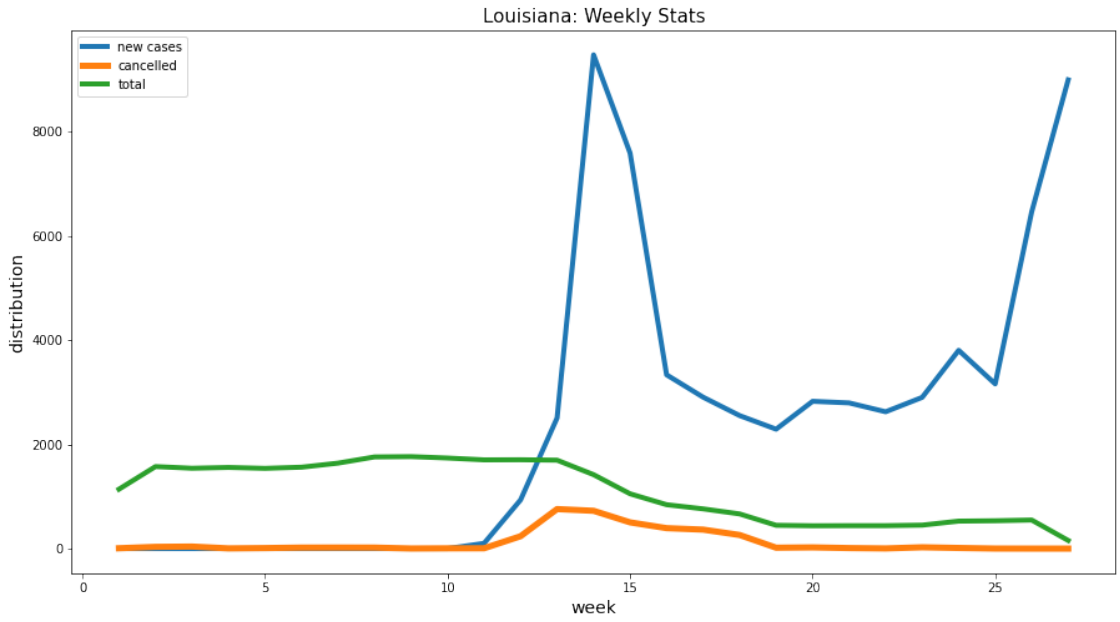
[115]:
```
<matplotlib.legend.Legend at 0x7fe389bef190>
```



[116]:
```
fig = plt.figure(figsize=(15,8))
plt.xlabel('week', fontsize=14)
plt.ylabel('distribution', fontsize=14)
x = np.linspace(0, 3000)

plt.plot(df_covid_flights_la['week'], df_covid_flights_la['new_case'],␣
 ↪label="new cases", linewidth = 4)
plt.plot(df_covid_flights_la['week'], df_covid_flights_la['CANCELLED'],␣
 ↪label="cancelled", linewidth = 5)
plt.plot(df_covid_flights_la['week'], df_covid_flights_la['total'],␣
 ↪label="total", linewidth = 4)
plt.title("Louisiana: Weekly Stats", fontsize = 15)
plt.legend(fontsize = 10)
```

[116]:
```
<matplotlib.legend.Legend at 0x7fe38a7239d0>
```



[119]:
```
!sudo apt-get install texlive-xetex texlive-fonts-recommended␣
 ↪texlive-plain-generic &
!jupyter nbconvert --to pdf /content/drive/Shareddrives/CSE544_Project/
 ↪Exploratory/Inference-1.ipynb
```

```
Reading package lists… Done
Building dependency tree
Reading state information… Done
texlive-fonts-recommended is already the newest version (2017.20180305-1).
```

```
texlive-plain-generic is already the newest version (2017.20180305-2).
texlive-xetex is already the newest version (2017.20180305-1).
The following packages were automatically installed and are no longer required:
  libnvidia-common-460 nsight-compute-2020.2.0
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 42 not upgraded.
[NbConvertApp] WARNING | pattern
'/content/drive/Shareddrives/CSE544_Project/flight_dataset/Inference-1.ipynb'
matched no files
This application is used to convert notebook files (*.ipynb)
        to various other formats.

        WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.

Options
=======
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
To see all configurable class-options for some <cmd>, use:
    <cmd> --help-all

--debug
    set log level to logging.DEBUG (maximize logging output)
    Equivalent to: [--Application.log_level=10]
--show-config
    Show the application's configuration (human-readable format)
    Equivalent to: [--Application.show_config=True]
--show-config-json
    Show the application's configuration (json format)
    Equivalent to: [--Application.show_config_json=True]
--generate-config
    generate default config file
    Equivalent to: [--JupyterApp.generate_config=True]
-y
    Answer yes to any questions instead of prompting.
    Equivalent to: [--JupyterApp.answer_yes=True]
--execute
    Execute the notebook prior to export.
    Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
    Continue notebook execution even if one of the cells throws an error and
include the error message in the cell output (the default behaviour is to abort
conversion). This flag is only relevant if '--execute' was specified, too.
    Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
    read a single notebook file from stdin. Write the resulting notebook with
default basename 'notebook.*'
    Equivalent to: [--NbConvertApp.from_stdin=True]
```

```
--stdout
    Write notebook output to stdout instead of files.
    Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
    Run nbconvert in place, overwriting the existing notebook (only
            relevant when converting to notebook format)
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=]
--clear-output
    Clear output of current file and save in place,
            overwriting the existing notebook.
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=
--ClearOutputPreprocessor.enabled=True]
--no-prompt
    Exclude input and output prompts from converted document.
    Equivalent to: [--TemplateExporter.exclude_input_prompt=True
--TemplateExporter.exclude_output_prompt=True]
--no-input
    Exclude input cells and output prompts from converted document.
            This mode is ideal for generating code-free reports.
    Equivalent to: [--TemplateExporter.exclude_output_prompt=True
--TemplateExporter.exclude_input=True]
--log-level=<Enum>
    Set the log level by value or name.
    Choices: any of [0, 10, 20, 30, 40, 50, 'DEBUG', 'INFO', 'WARN', 'ERROR',
'CRITICAL']
    Default: 30
    Equivalent to: [--Application.log_level]
--config=<Unicode>
    Full path of a config file.
    Default: ''
    Equivalent to: [--JupyterApp.config_file]
--to=<Unicode>
    The export format to be used, either one of the built-in formats
            ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook',
'pdf', 'python', 'rst', 'script', 'slides']
            or a dotted object name that represents the import path for an
            `Exporter` class
    Default: 'html'
    Equivalent to: [--NbConvertApp.export_format]
--template=<Unicode>
    Name of the template file to use
    Default: ''
    Equivalent to: [--TemplateExporter.template_file]
--writer=<DottedObjectName>
    Writer class used to write the
                                        results of the conversion
```

```
    Default: 'FilesWriter'
    Equivalent to: [--NbConvertApp.writer_class]
--post=<DottedOrNone>
    PostProcessor class used to write the
                                        results of the conversion
    Default: ''
    Equivalent to: [--NbConvertApp.postprocessor_class]
--output=<Unicode>
    overwrite base name use for output files.
                can only be used when converting one notebook at a time.
    Default: ''
    Equivalent to: [--NbConvertApp.output_base]
--output-dir=<Unicode>
    Directory to write output(s) to. Defaults
                                    to output to the directory of each notebook.
To recover
                                    previous default behaviour (outputting to the
current
                                    working directory) use . as the flag value.
    Default: ''
    Equivalent to: [--FilesWriter.build_directory]
--reveal-prefix=<Unicode>
    The URL prefix for reveal.js (version 3.x).
            This defaults to the reveal CDN, but can be any url pointing to a
copy
            of reveal.js.
            For speaker notes to work, this must be a relative path to a local
            copy of reveal.js: e.g., "reveal.js".
            If a relative path is given, it must be a subdirectory of the
            current directory (from which the server is run).
            See the usage documentation
            (https://nbconvert.readthedocs.io/en/latest/usage.html#reveal-js-
html-slideshow)
            for more details.
    Default: ''
    Equivalent to: [--SlidesExporter.reveal_url_prefix]
--nbformat=<Enum>
    The nbformat version to write.
            Use this to downgrade notebooks.
    Choices: any of [1, 2, 3, 4]
    Default: 4
    Equivalent to: [--NotebookExporter.nbformat_version]

Examples
--------

    The simplest way to use nbconvert is
```

```
            > jupyter nbconvert mynotebook.ipynb

            which will convert mynotebook.ipynb to the default format (probably
HTML).

            You can specify the export format with `--to`.
            Options include ['asciidoc', 'custom', 'html', 'latex', 'markdown',
'notebook', 'pdf', 'python', 'rst', 'script', 'slides'].

            > jupyter nbconvert --to latex mynotebook.ipynb

            Both HTML and LaTeX support multiple output templates. LaTeX
includes
            'base', 'article' and 'report'.  HTML includes 'basic' and 'full'.
You
            can specify the flavor of the format used.

            > jupyter nbconvert --to html --template basic mynotebook.ipynb

            You can also pipe the output to stdout, rather than a file

            > jupyter nbconvert mynotebook.ipynb --stdout

            PDF is generated via latex

            > jupyter nbconvert mynotebook.ipynb --to pdf

            You can get (and serve) a Reveal.js-powered slideshow

            > jupyter nbconvert myslides.ipynb --to slides --post serve

            Multiple notebooks can be given at the command line in a couple of
            different ways:

            > jupyter nbconvert notebook*.ipynb
            > jupyter nbconvert notebook1.ipynb notebook2.ipynb

            or you can specify the notebooks list in a config file, containing::

                c.NbConvertApp.notebooks = ["my_notebook.ipynb"]

            > jupyter nbconvert --config mycfg.py

To see all available configurables, use `--help-all`.
```

[ ]: