Preliminary setup

\* Create a directory named “deloitte\_hackathon”

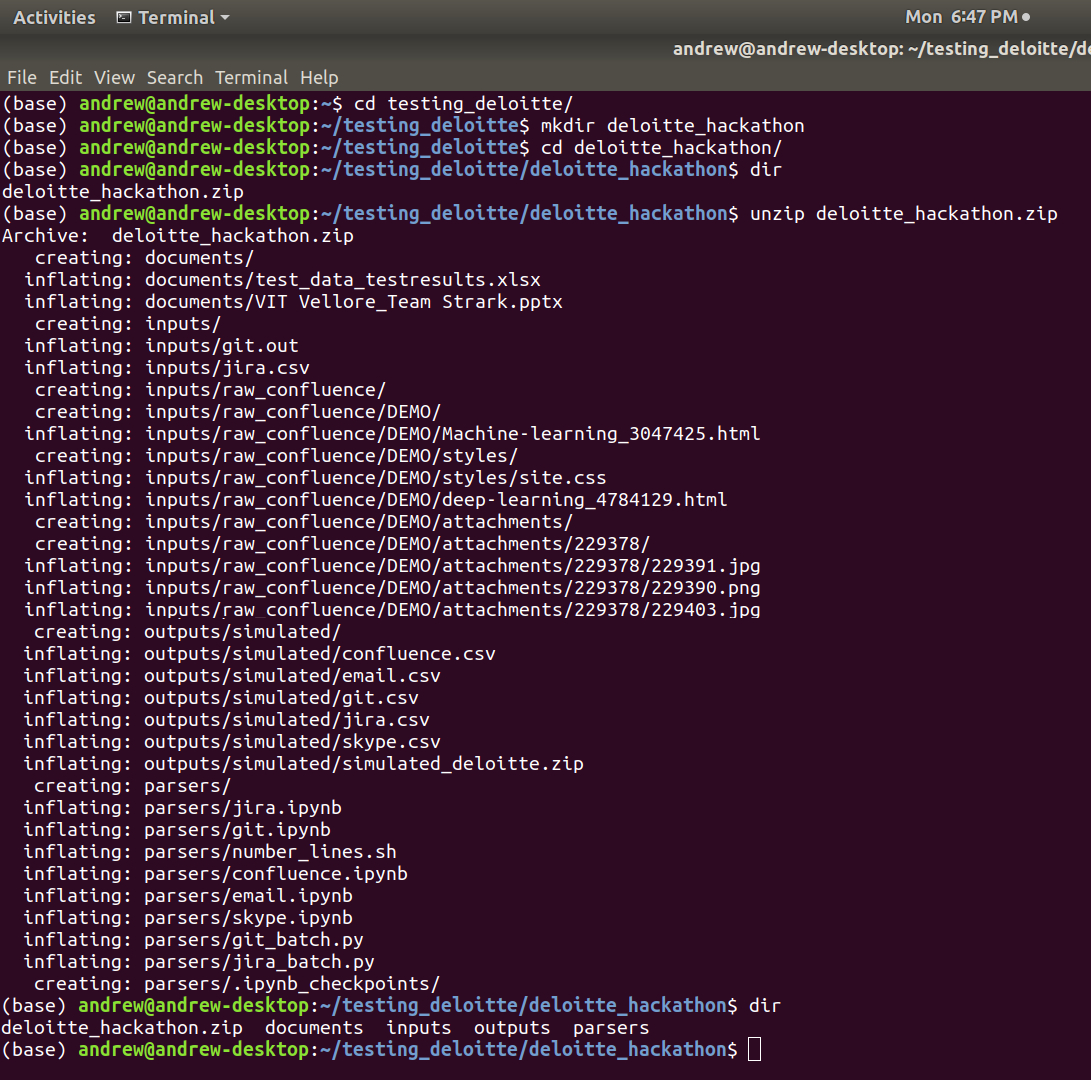
$ mkdir deloitte\_hackathon

\* Place the given source zip file (deloitte\_hackathon.zip) in the newly created directory.

\* Change directory to the deloitte\_hackathon and unzip the source zip file.

$ cd deloitte\_hackathon

$ unzip deloitte\_hackathon.zip



Common Prerequisites:

1. Python3

2. Pandas

3. Jupyter notebook

1. To collect information from a git repository

Prerequisite: Git tool

\* Copy and place the shell script (parsers/**number\_lines.sh**) in any git repository. E.g. <https://github.com/arzzen/git-quick-stats.git> (clone it locally using ‘git’).

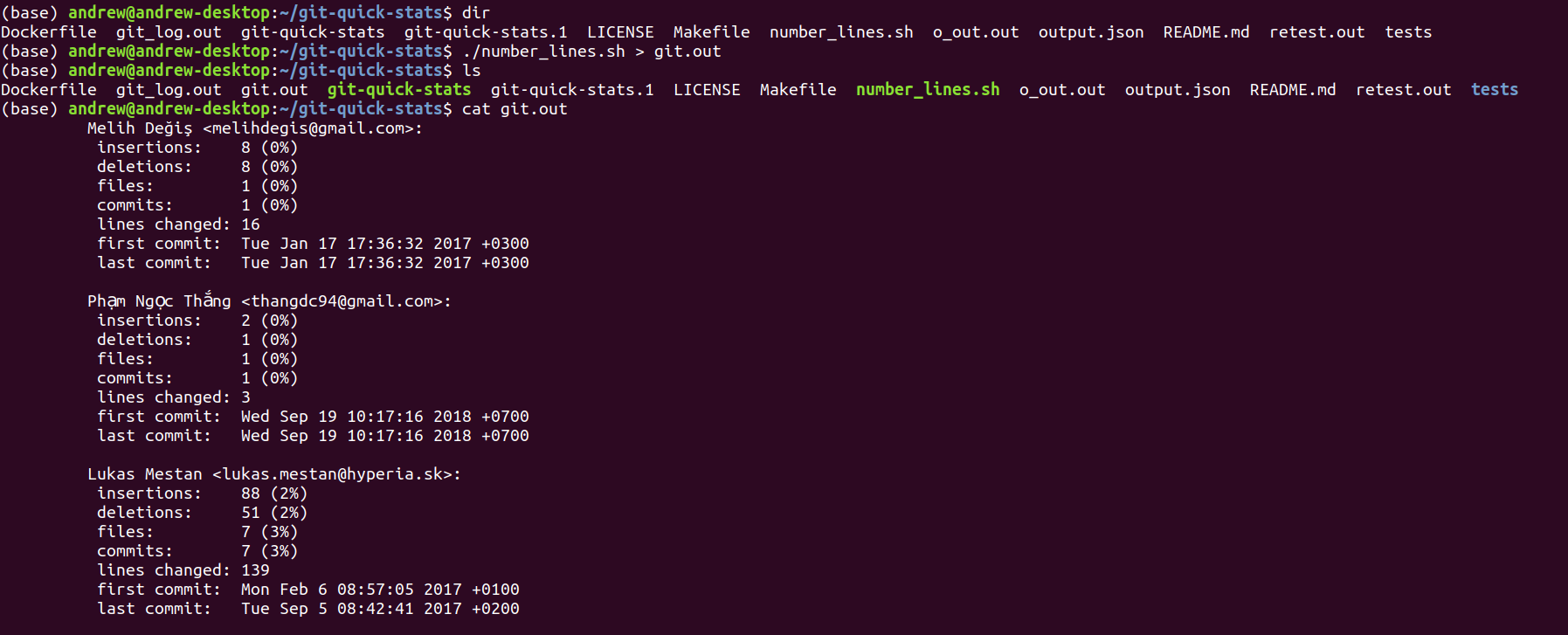
\* Execute the shell script from the git repository’s root directory.

e.g.

$ cd git-quick-stats

$ ./number\_lines.sh > *output\_file\_name.out*

e.g. : $ ./number\_lines.sh > git.out



\* The unparsed raw information would be stored in the output file in the root directory of the repository.

\* Copy and place the newly generated file in (deloitte\_hackathon/inputs/.)

\* Execute the (parsers/git\_batch.py) with first first argument as the newly generated file (git.out).

$ cd parsers

$ python git\_batch.py ../inputs/git.out



\* The output of this script will be stored in (deloitte\_hackatohn/outputs/git.csv). This will be the final output of the parser. It contains how many lines were modified by each and every contributor of that given repository.

2. To collect information from a jira project

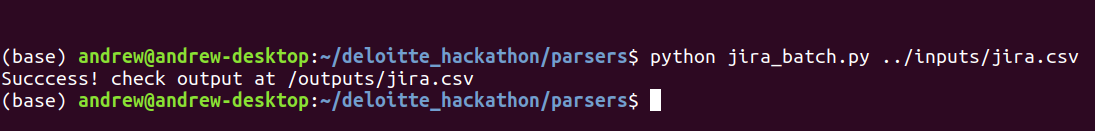
Prerequisite: Jira project

\* Export the Jira tickets of a project in csv file and place it in (deloitte\_hackathon/inputs/.). E.g. file is present under the name (inputs/**jira.csv**).

\* Execute the (parsers/jira\_batch.py) with first argument as the jira raw exported file.

$ cd parsers

$ python jira\_batch.py ../inputs/jira.csv



\* The output will be stored at *deloitte\_hackathon/outputs/jira.csv*. It contains the aggregated sum of tickets resolved by individual authors.

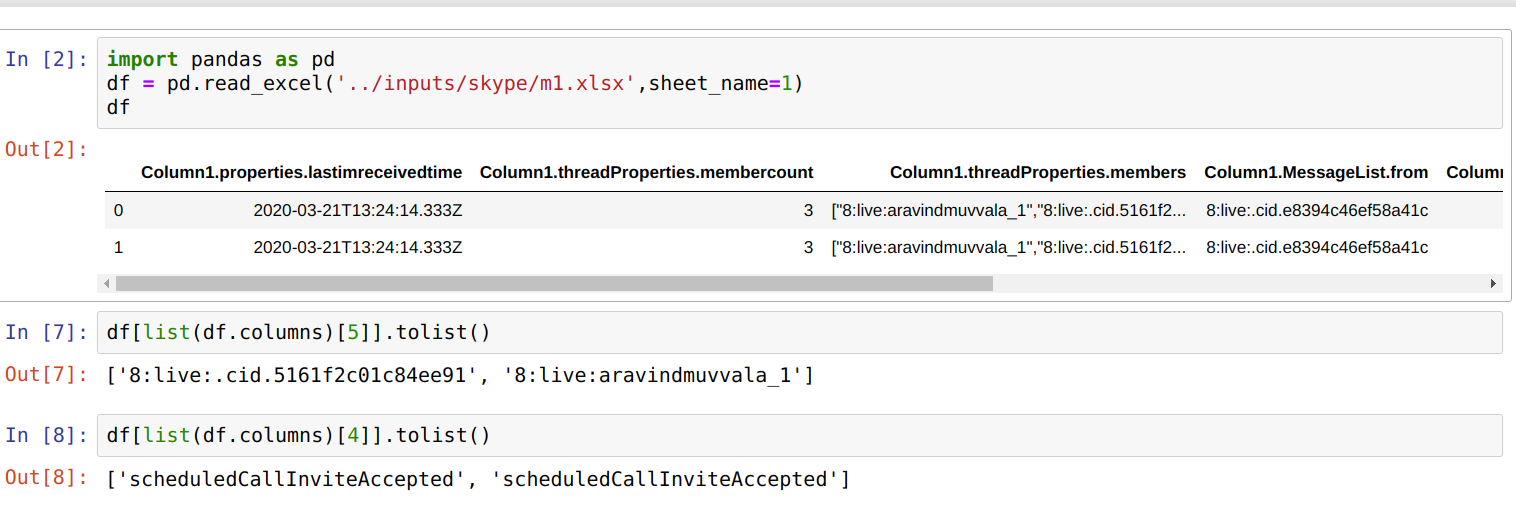
3. To collect information from skype call

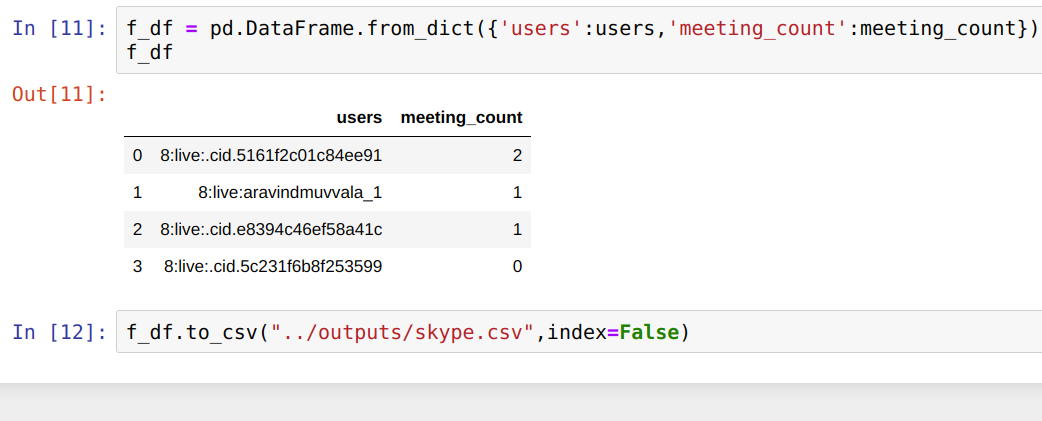
Prerequisite: Skype meetings

\* Export the skype call log and place the raw xlsx file in (deloitte\_hackathon/inputs/skype/.) Note: Sample files are given at the same directory

\* Execute the jupyternotebook (parsers/skype.ipynb) from the same directory.

\* The parsed output will be stored at *deloitte\_hackathon/outputs/skype.csv.* It contains the aggregated sum of meetings attended by individual participants.





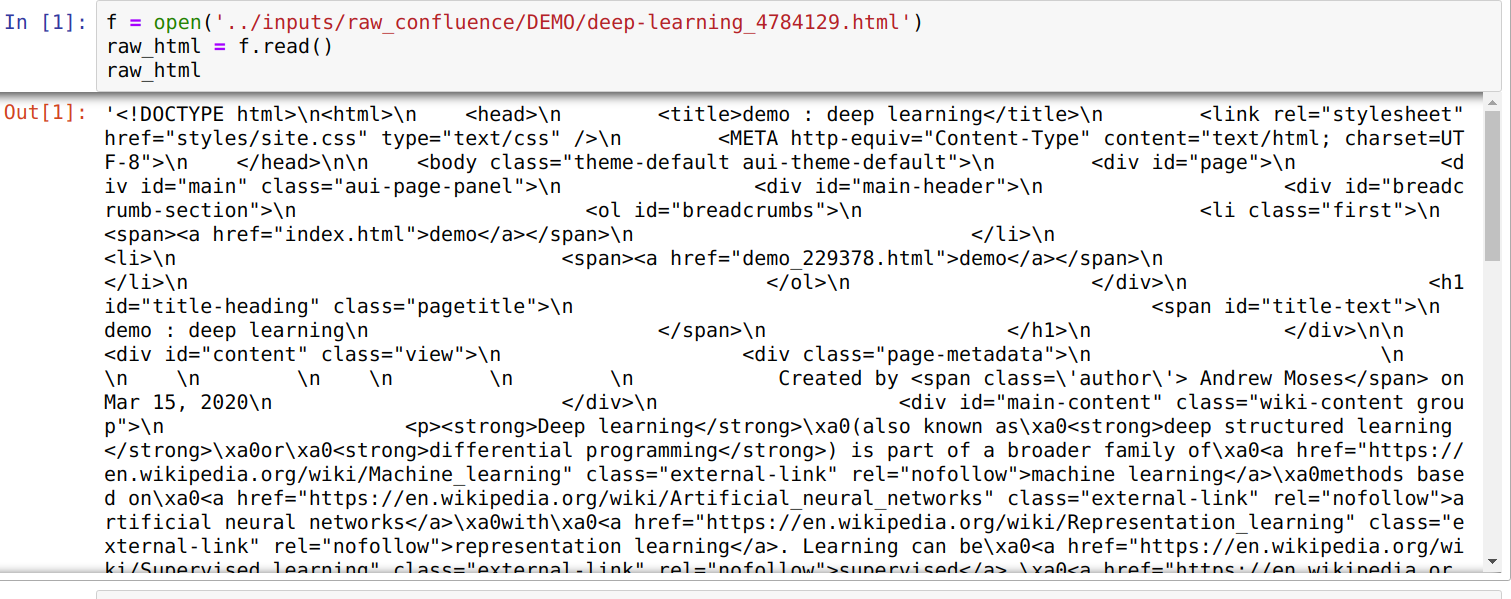
4. To collect information from confluence

Prerequisite: Confluence project

\* Export the confluence project from project settings. Export in HTML format. And place the files in (deloitte\_hackathon/inputs/raw\_confluence/.) Note: Sample files are given at the same directory.

\* Execute the jupyternotebook (parsers/confluence.ipynb) from the same directory.

\* The parsed output will be stored at *deloitte\_hackathon/outputs/confluence.csv* It contains the aggregate sum of character count contributions by individual authors.





5. To collect unstructured email insights

\* For this use case, a subset of open Enron email dataset was utilized. The input file is given at deloitte\_hackathon/inputs/email\_1k.csv It holds 1000 records (subset) of original emails collected from Enron company (opensourced). <https://www.cs.cmu.edu/~enron/>

\* Execute the jupyternotebook (parsers/email.ipynb) from the same directory.

\* Unsupervised learning was followed and emails were clustered. Associates who’s emails belong to a region of interest (ROI) cluster will be awarded bonus points.

\* The parsed output will be stroed at *deloitte\_hackathon/outputs/email.csv* It contains the bonus points awarded to individual email ids.

