1. Introduction

Version control repositories like CVS, Subversion or Git can be a real gold mine for software developers. They contain every change to the source code including the date (the "when"), the responsible developer (the "who"), as well as little message that describes the intention (the "what") of a change.

(https://commons.wikimedia.org/wiki/File:Tux.svg)

In this notebook, we will analyze the evolution of a very famous open-source project – the Linux kernel. The Linux kernel is the heart of some Linux distributions like Debian, Ubuntu or CentOS.

We get some first insights into the work of the development efforts by

- identifying the TOP 10 contributors and
- visualizing the commits over the years.

Linus Torvalds, the (spoiler alert!) main contributor to the Linux kernel (and also the creator of Git), created a mirror of the Linux repository on GitHub (https://github.com/torvalds/linux/). It contains the complete history of kernel development for the last 13 years.

For our analysis, we will use a Git log file with the following content:

In [1]:
```
# Printing the content of git_log_excerpt.csv
print(open('datasets/git_log_excerpt.csv'))
```

```
<_io.TextIOWrapper name='datasets/git_log_excerpt.csv' mode='r' en
coding='UTF-8'>
```

2. Reading in the dataset

The dataset was created by using the command `git log --encoding=latin-1 --pretty="%at#%aN"`. The `latin-1` encoded text output was saved in a header-less csv file. In this file, each row is a commit entry with the following information:

- `timestamp`: the time of the commit as a UNIX timestamp in seconds since 1970-01-01 00:00:00 (Git log placeholder `"%at"`)
- `author`: the name of the author that performed the commit (Git log placeholder `"%aN"`)

The columns are separated by the number sign #. The complete dataset is in the `datasets/` directory. It is a `gz`-compressed csv file named `git_log.gz`.

In [2]:
```
# Loading in the pandas module
import pandas as pd

# Reading in the log file
git_log = pd.read_csv('datasets/git_log.gz', sep='#', encoding='lat
in-1', header=None, names=['timestamp', 'author'])

# Printing out the first 5 rows
print(git_log.head())
```

```
    timestamp          author
0  1502826583  Linus Torvalds
1  1501749089   Adrian Hunter
2  1501749088   Adrian Hunter
3  1501882480       Kees Cook
4  1497271395       Rob Clark
```

3. Getting an overview

The dataset contains the information about every single code contribution (a "commit") to the Linux kernel over the last 13 years. We'll first take a look at the number of authors and their commits to the repository.

In [3]:
```
# calculating number of commits
number_of_commits = git_log['timestamp'].count()

# calculating number of authors
number_of_authors = git_log['author'].nunique()

# printing out the results
print("%s authors committed %s code changes." % (number_of_authors,
number_of_commits))
```

```
17385 authors committed 699071 code changes.
```

4. Finding the TOP 10 contributors

There are some very important people that changed the Linux kernel very often. To see if there are any bottlenecks, we take a look at the TOP 10 authors with the most commits.

In [4]:
```
# Identifying the top 10 authors
top_10_authors = git_log['author'].value_counts()[:10]

# Listing contents of 'top_10_authors'
top_10_authors
```

Out[4]:
```
Linus Torvalds           23361
David S. Miller           9106
Mark Brown                6802
Takashi Iwai              6209
Al Viro                   6006
H Hartley Sweeten         5938
Ingo Molnar               5344
Mauro Carvalho Chehab     5204
Arnd Bergmann             4890
Greg Kroah-Hartman        4580
Name: author, dtype: int64
```

5. Wrangling the data

For our analysis, we want to visualize the contributions over time. For this, we use the information in the `timestamp` column to create a time series-based column.

In [5]:
```
# converting the timestamp column
git_log['timestamp'] = pd.to_datetime(git_log['timestamp'],unit='s'
)

# summarising the converted timestamp column
git_log['timestamp'].describe()
```

Out[5]:
```
count                  699071
unique                 668448
top       2008-09-04 05:30:19
freq                       99
first     1970-01-01 00:00:01
last      2037-04-25 08:08:26
Name: timestamp, dtype: object
```

6. Treating wrong timestamps

As we can see from the results above, some contributors had their operating system's time incorrectly set when they committed to the repository. We'll clean up the `timestamp` column by dropping the rows with the incorrect timestamps.

In [6]:
```
# determining the first real commit timestamp
first_commit_timestamp = git_log['timestamp'].iloc[-1]

# determining the last sensible commit timestamp
last_commit_timestamp = pd.to_datetime('today')

# filtering out wrong timestamps
corrected_log = git_log[(git_log['timestamp']>= first_commit_timest
amp)&(git_log['timestamp']<=last_commit_timestamp)]

# summarising the corrected timestamp column
corrected_log.describe()
```

Out[6]:

	timestamp	author
count	698569	698568
unique	667977	17375
top	2008-09-04 05:30:19	Linus Torvalds
freq	99	23361
first	2005-04-16 22:20:36	NaN
last	2017-10-03 12:57:00	NaN

7. Grouping commits per year

To find out how the development activity has increased over time, we'll group the commits by year and count them up.

In [7]:
```
# Counting the no. commits per year
commits_per_year = corrected_log.groupby(pd.Grouper(key='timestamp'
, freq='AS')).count()
commits_per_year.rename(columns={'author': '# of commits'}, inplace
=True)

# Listing the first rows
commits_per_year.head()
```

Out[7]:

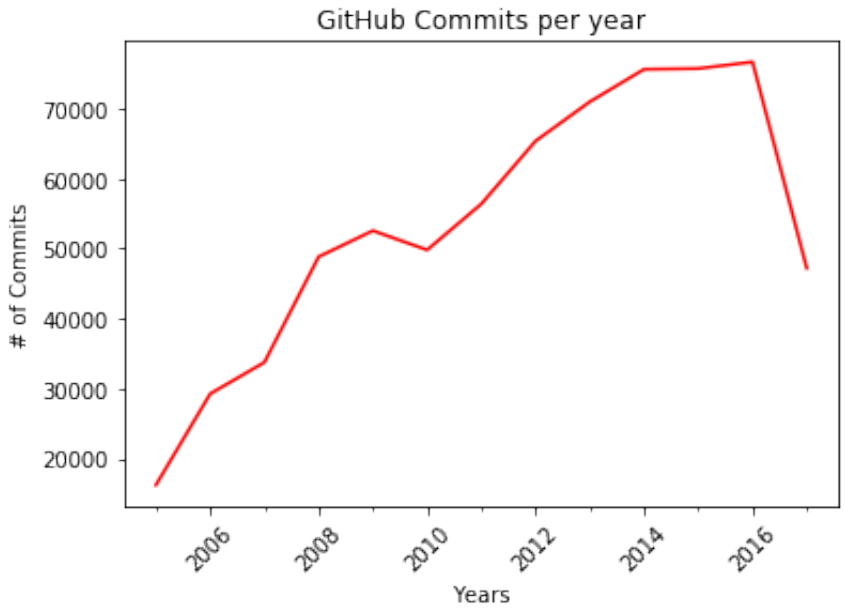
	# of commits
timestamp	
2005-01-01	16229
2006-01-01	29255
2007-01-01	33759
2008-01-01	48847
2009-01-01	52572

8. Visualizing the history of Linux

Finally, we'll make a plot out of these counts to better see how the development effort on Linux has increased over the the last few years.

In [8]:
```
# Setting up plotting in Jupyter notebooks
import matplotlib.pyplot as plt
%matplotlib inline

# plot the data
commits_per_year.plot(title='GitHub Commits per year', legend=False
, color='r')
plt.xlabel('Years')
plt.ylabel('# of Commits')
plt.xticks(rotation=45)
plt.show()
```



9. Conclusion

Thanks to the solid foundation and caretaking of Linux Torvalds, many other developers are now able to contribute to the Linux kernel as well. There is no decrease of development activity at sight!

In [9]:
```
# calculating or setting the year with the most commits to Linux
year_with_most_commits = ...
```