Alexandre Castera - Rafaël de Lavergne - Stefan de Lavergne - Antoine Guyot - Oliver Straub

Data Science Project Report Master X-ITIE 2016-2017

Business Intelligence for Venture Capital Firms

The goal of this project is to analyse trends in the creation of startups and Venture Capital investments in various markets worldwide. Several types of data shall be used during this project: news articles from Techcrunch and investments data from Crunchbase. The ambition is to be able to extract powerful market intelligence to:

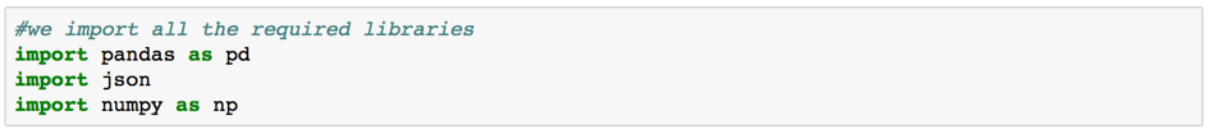
- Understand why startups fail
- Know what key differentiations are important for VCs
- Identify markets currently being disrupted & tomorrow's opportunities
- Map the VC ecosystem with relationships & targeted markets

1. <u>Data Pre-Processing</u>

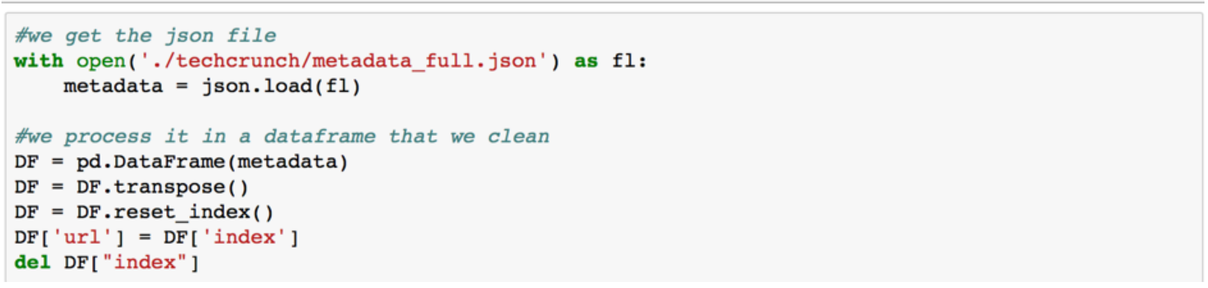
The first part of the project was to format the raw tech crunch data, in order to be able to analyse the data with Anseri, a cutting edge text analytics technology.

We got the data in two parts. On the one hand, a big JSON file containing all the metadata of Techcrunch articles and on the other hand a folder gathering all the articles contents as .txt files. The data came in form of 15.049 separate .txt files which contain the news messages. Furthermore, information like author, filename, links, time, title and the url were included.

To be able to process complex algorithms on both data and metadata, we decided to gather everything in a single Dataframe using the pandas library

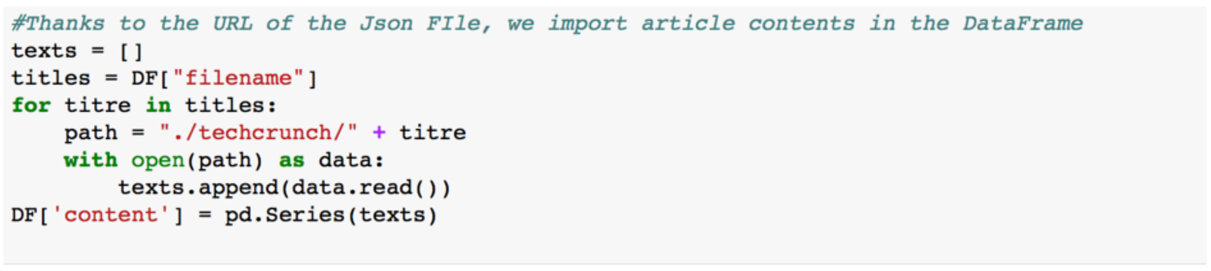


```python
#we import all the required libraries
import pandas as pd
import json
import numpy as np
```



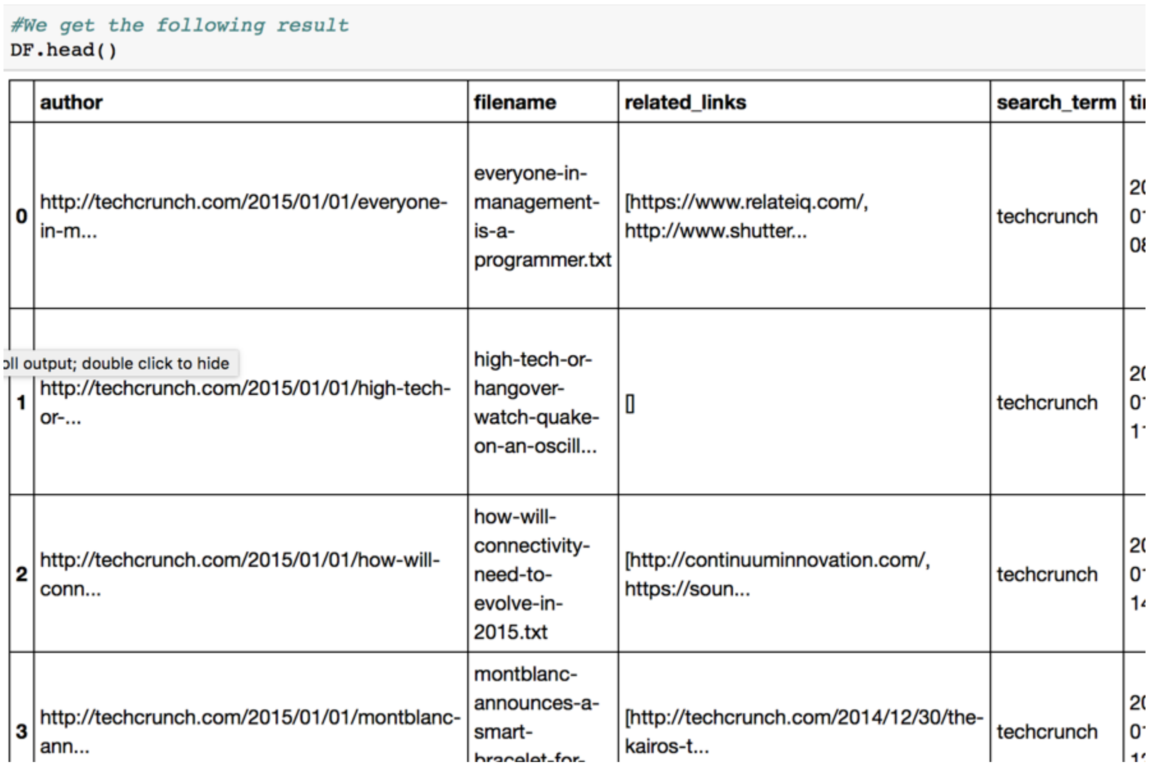
```python
#we get the json file
with open('./techcrunch/metadata_full.json') as fl:
    metadata = json.load(fl)

#we process it in a dataframe that we clean
DF = pd.DataFrame(metadata)
DF = DF.transpose()
DF = DF.reset_index()
DF['url'] = DF['index']
del DF["index"]
```



```python
#Thanks to the URL of the Json FIle, we import article contents in the DataFrame
texts = []
titles = DF["filename"]
for titre in titles:
    path = "./techcrunch/" + titre
    with open(path) as data:
        texts.append(data.read())
DF['content'] = pd.Series(texts)
```

Is our code efficient? It is probably optimal since we don't have redundant operations and all operations are necessary. In addition we only do effective operations. First, during the Json process we only read the file and clean the data without using expensive functions. (The only ones we use are built-in and written in C language for Pandas). Likewise we use a python list, which is a very light object, to read the TechCrunch articles. We turn it into a Series only at the end so we don't use to much of the Pandas functions which could me heavier.



```
#We get the following result
DF.head()
```

	author	filename	related_links	search_term	ti
0	http://techcrunch.com/2015/01/01/everyone-in-m...	everyone-in-management-is-a-programmer.txt	[https://www.relateiq.com/, http://www.shutter...	techcrunch	2 0 0
1	http://techcrunch.com/2015/01/01/high-tech-or-...	high-tech-or-hangover-watch-quake-on-an-oscill...	[]	techcrunch	2 0 1
2	http://techcrunch.com/2015/01/01/how-will-conn...	how-will-connectivity-need-to-evolve-in-2015.txt	[http://continuuminnovation.com/, https://soun...	techcrunch	2 0 1
3	http://techcrunch.com/2015/01/01/montblanc-ann...	montblanc-announces-a-smart-	[http://techcrunch.com/2014/12/30/the-kairos-t...	techcrunch	2 0

2. Using Anseri

2.1. Formatting for Anseri

The Dataframe still contains irrelevant information though. Therefore, all columns besides “timestamp”, “title” and “content” are dropped.

```
#reduce to relevant info only
df.drop('author', axis=1, inplace=True)
df.drop('filename', axis=1, inplace=True)
df.drop('related_links', axis=1, inplace=True)
df.drop('search_term', axis=1, inplace=True)
df.drop('url', axis=1, inplace=True)
df.head()[0:1]
```

	timestamp	title	content
0	2015-01-01 08:00:04	Everyone In Management Is A Programmer	Editors note:Adam Evans is the co-founder and ...

The timestamp column holds the date and the exact time. The latter being removed by a string split.

```
df['timestamp'] = df['timestamp'].apply(lambda x: str(x).split()[0])
```

```
df.head()[0:4]
```

	timestamp	title	content
0	2015-01-01	Everyone In Management Is A Programmer	Editors note:Adam Evans is the co-founder and ...
1	2015-01-01	High Tech Or Hangover? Watch Quake On An Oscil...	Head spinning? Stomach gurgling? Did you just ...
2	2015-01-01	How Connectivity Will Need To Evolve	Editors note:Kevin Young is the senior vice pr...
3	2015-01-01	Montblanc Announces A Smart Bracelet For Your ...	What did I tell you? In a last-ditch effort to...

In the following step, we are creating a dictionary from the data frame, with a fixed timestamp.

```
contentcols = ['title','content']
splitter = ai.data.database_controller.SentenceSplitter(contentcols)

def fix_timestamps(source):
    for row in source:
        row['timestamp'] = int(row['timestamp'])
        yield row

dict_techcrunch = [{'timestamp':i[1].timestamp,'title':i[1].title,'content':i[1].content } for i in df.iterrows()]
```

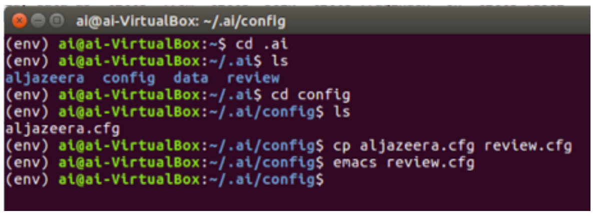
Creating the config file

```
dbname = 'techcrunch'
dbpath = '~/.ai/data/{}.db'.format(dbname)
contentcols = ['title','content']
timecol = 'timestamp'

#need to create a dictionary based on the pandas dataframe
ai.data.backends.sqlite.SQLiteController_FTS.from_iterable(iterable=dict_techcrunch, dbpath=dbpath,
                                                            contentcols=contentcols, timecol=timecol)
```

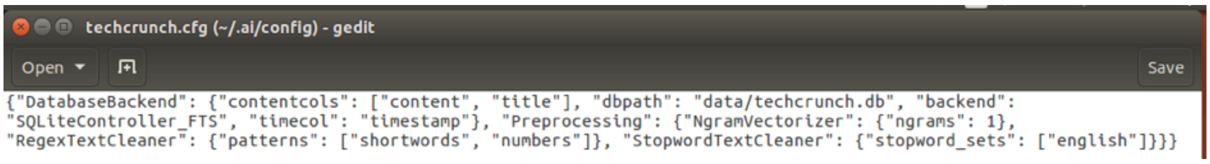
First, we are creating a config file, by copying the already existing “aljazeera.cfg” configuration file and rename it to “techcrunch.cfg”. The following table shows the commands to edit the config file for both systems, iOS and Linux.

For iOS:	For Linux:



```
ai@ai-VirtualBox: ~/.ai/config
(env) ai@ai-VirtualBox:~$ cd .ai
(env) ai@ai-VirtualBox:~/.ai$ ls
aljazeera  config  data  review
(env) ai@ai-VirtualBox:~/.ai$ cd config
(env) ai@ai-VirtualBox:~/.ai/config$ ls
aljazeera.cfg
(env) ai@ai-VirtualBox:~/.ai/config$ cp aljazeera.cfg review.cfg
(env) ai@ai-VirtualBox:~/.ai/config$ emacs review.cfg
(env) ai@ai-VirtualBox:~/.ai/config$
```

Then we have to change its parameters depending on the focus of our analysis (see next figure). Once created, the config file can be kept as long as the variables are not changed for another analysis.



```
{"DatabaseBackend": {"contentcols": ["content", "title"], "dbpath": "data/techcrunch.db", "backend":
"SQLiteController_FTS", "timecol": "timestamp"}, "Preprocessing": {"NgramVectorizer": {"ngrams": 1},
"RegexTextCleaner": {"patterns": ["shortwords", "numbers"]}, "StopwordTextCleaner": {"stopword_sets": ["english"]}}}
```

To verify the time period of the covered data, we displayed the time range (with d = ai.Dataset("techcrunch")).

```
print(d.time_range)
('2015-01-01', '2016-06-06')
```

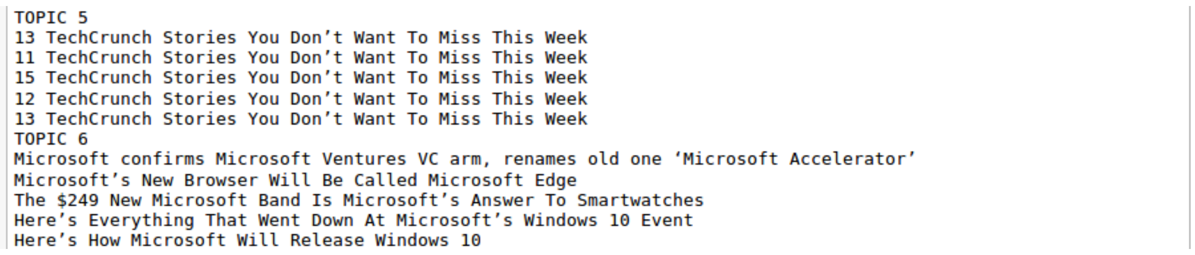
2.2. Using Anseri

After having set up Anseri, the data analysis can be started. At this stage, we tested Anseri's functions in order to get familiar with its possibilities, before answering the questions at hand. The shown figures are therefore over the whole time period of the data without any elimination of unnecessary words. Due to its wide range of options, only some of Anseri's functions are shown in the following.

```
print(topics)    # TopicModel object has a convenient representation
apple watch app: [apple, watch, app, iphone, music, ios, store, pay]
---
raises startup series: [raises, startup, series, platform, service, funding, led, mobile]
---
google launches android: [google, launches, android, search, apps, play, cloud, photos]
---
facebook tech video: [facebook, tech, video, messenger, live, users, ads, news]
---
techcrunch week stories: [techcrunch, week, stories, disrupt, podcast, bitcoin, tickets, london]
---
microsoft windows data: [microsoft, windows, data, build, big, twitter, office, surface]
---
```

The first set of words represent the strongest portion of the topic, covering approximately 95% of the topic strength. The words in brackets represent the total list of words defining the topic. Taking the above figure as an example, it becomes obvious that the main topic in the last months was related to “apple watch app”. The words in the brackets such as “iphone”, “ios”, “store” and “pay” are describing the topic in this case.

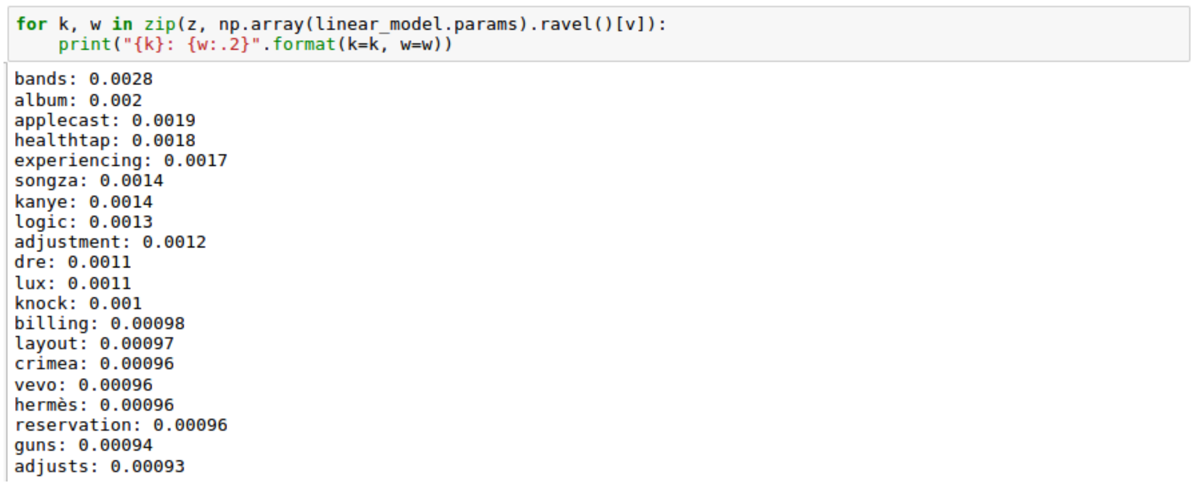
Furthermore, Anseri contains document recommendations commands, which display the strongest examples of each topic in a collection of topics.



```
TOPIC 5
13 TechCrunch Stories You Don’t Want To Miss This Week
11 TechCrunch Stories You Don’t Want To Miss This Week
15 TechCrunch Stories You Don’t Want To Miss This Week
12 TechCrunch Stories You Don’t Want To Miss This Week
13 TechCrunch Stories You Don’t Want To Miss This Week
TOPIC 6
Microsoft confirms Microsoft Ventures VC arm, renames old one ‘Microsoft Accelerator’
Microsoft’s New Browser Will Be Called Microsoft Edge
The $249 New Microsoft Band Is Microsoft’s Answer To Smartwatches
Here’s Everything That Went Down At Microsoft’s Windows 10 Event
Here’s How Microsoft Will Release Windows 10
```

The output of Topic 5 shows repetitive messages without significant information. It becomes obvious that we have to adjust the code and avoid repetitions in order to get a more relevant output.

Another powerful command in Anseri is the possibility to show all image words that are related to a specific search term. The following figure gives an example with the word “apple”.



```
for k, w in zip(z, np.array(linear_model.params).ravel()[v]):
    print("{k}: {w:.2}".format(k=k, w=w))
```

```
bands: 0.0028
album: 0.002
applecast: 0.0019
healthtap: 0.0018
experiencing: 0.0017
songza: 0.0014
kanye: 0.0014
logic: 0.0013
adjustment: 0.0012
dre: 0.0011
lux: 0.0011
knock: 0.001
billing: 0.00098
layout: 0.00097
crimea: 0.00096
vevo: 0.00096
hermès: 0.00096
reservation: 0.00096
guns: 0.00094
adjusts: 0.00093
```

Being familiar with Anseri’s function, the next step is to adapt the code to our market analysis questions, which are described in detail in the next chapter.

3. <u>What's next?</u>

We tried to define some goals for our next steps in this scientific project. The idea is to be in a research perspective, to start from the data and to see if by large analysis we can find some interesting staff. After this identification we will be able to narrow down and to go deeply in one topic.
This is the analysis that we want to try with the data:

- Find some evolution on topics on time basis. We have data from 2013 to 2016 so it could be interesting to monitor the evolution of promising topics such as VR and drones for example. This will let us understand how huge is the growth and the investment in those sectors.
- Find some differences based on countries. Since we have some data from different startups in different countries, we could provide analysis by country to know if the investment is done at the same time in every country or if countries such as the US are precursors in investment.
- Select 3 topics related to our personal projects (foodtech, fintech, medtech) and try to deeply analyze the market and to compare the startups. The idea is to use this scientific project to help us in our personal projects of startups. We could use those data to define the competitive framework and to understand the average investment in our sector.
- Find the VCs that have invested in different fields and try to find clusters to understand if there is or not a specialization. Indeed, some VCs are specialized in a sector and maybe we could do a rating of VCs as a function of their investment amount.
- Correlation between investment in unicorns and trends in startup investment in the sector. It could be interesting to see the impact of a huge investment in a startup on other startups from the same field. (We can think about the investment in Slack which triggered many investments in startups doing chatbots)

This is a very interesting project, we like that this is really open and that we can apply it to the markets we are interested in. There is a lot of work to be done in the next few months if we want to reach our goals and ambitions.