Example FIVE - Scrape & Enrich your data using TRAFILATURA

Introduction

On this example, we will use Trafilatura, a powerful text extractor on a list of URLs.

We will use a dataset which is an extract of blog posts urls from a francophone conspiracist website.

Our scenario implies to retrieve the articles, and then analyse tjem with TextRazor, an online semantic extractor, in order to analyse the topics of these articles.

SUM-UP of the different steps

- Import a list of URLs
- Scrape the content using Trafilatura
- Extract entities using TextRazor API

Pre-requisite : a free account on TextRazor

Go to https://www.textrazor.com/, create a free account. It will give you access to an API key that allows you 500 requests per day for free.

Import the data

- create a new project in OpenRefine by importing the csv file called `moutons.csv`.

- create a new column based on the url column with the trafilatura command. `"trafilatura -u "+value`

Important You can add a layer of anonymization, by using the *torify* command, which will wrap the command into TOR. Instead of `"trafilatura -u "+value`, simply use `"torify trafilatura -u "+value`.

- Create a new column based on this column, a Jython script :

The jython script that we use is the following :

```
import time
import commands
import random
# get status and output of the command
status, output = commands.getstatusoutput(value)
# add a random between 0 and 1s pause to avoid ddos on servers... Be kind
to APIs!
time.sleep(random.randint(0, 1))
# returns the result of the command
return output.decode("utf-8")
```

Done!

you get a column with the text of every article.

Create a copy of this column

We will work on the copy of the column

Clean a little bit the text

- Remove the unused space character by doing `cell->common transform->collapse consecutive whitespace`
- Replace some characters which may cause issues : `value.replace(/@|\+|#|\?|&|=|\//,"")`
- Remove emojis.

Use the transform function in Jython :

```
import re

def remove_emojis(data):
    emoj = re.compile("["
        u"\U0001F600-\U0001F64F"  # emoticons
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
        u"\U00002500-\U00002BEF"  # chinese char
        u"\U00002702-\U000027B0"
        u"\U00002702-\U000027B0"
        u"\U000024C2-\U0001F251"
        u"\U0001f926-\U0001f937"
        u"\U00010000-\U0010ffff"
        u"\u2640-\u2642"
        u"\u2600-\u2B55"
        u"\u200d"
        u"\u23cf"
        u"\u23e9"
        u"\u231a"
        u"\ufe0f"  # dingbats
        u"\u3030"
                      "]+", re.UNICODE)
    return re.sub(emoj, '', data)

return remove_emojis(value)
```

Enrich the data with TextRazor

The TextRazor API requires a POST request instead of a GET request. You cannot simply craft a URL to access it; it's mandatory to pas some parameter via a CURL command.

```
curl -s -X POST -H "x-textrazor-key: YOUR_API_KEY_HERE" -d
"extractors=entities" -d "text=YOUR_TEXT_TO_BE_ANALIZED_HERE"
https://api.textrazor.com/
```

In OpenRefine we we'll first need backslashed doble-quotes. We will use this code :

```
curl -s -X POST -H \"x-textrazor-key: YOUR_API_KEY_HERE\" -d
\"extractors=entities\" -d \"text=YOUR_TEXT_TO_BE_ANALIZED_HERE\"
https://api.textrazor.com/
```

Create a column based on the cleansed column using this GREL expression and replace YOUR_API_KEY_HERE by your real API key:

```
"curl -s -X POST -H \"x-textrazor-key: YOUR_API_KEY_HERE\" -d
\"extractors=entities\" -d \"text="+value+" \" https://api.textrazor.com/"
```

Then create a new columm based on this column, using the Jython script.

You will get from TextRazor the list of entities in your blog posts.

Extract entities

To extract the text, we will simply parse the JSON response like this :

```
forEach(value.parseJson().response.entities,v,v.entityEnglishId).sort().uni
ques().join(',')
```

This loop formula will retrieve every unique entity in the text.