**Keyword Detection on Websites**

Download Datasets

*This data project has been used as a take-home assignment in the recruitment process for the data science positions at PeakData.*

**Assignment**

Your task is to create an algorithm, that takes html page as input and infers if the page contains the information about **cancer tumorboard** or not. What is a tumor board? Tumor Board is a consilium of doctors (usually from different disciplines) discussing cancer cases in their departments. If you want to know more please read this [article](https://www.cancer.net/blog/2017-07/what-tumor-board-expert-qa).

The expected result is a CSV file for test data with columns [**doc\_id** and **prediction**].

*Bonus*: if you would like to go the extra mile in this task try to identify tumor board types **interdisciplinary**, **breast**, and any third type of tumor board up to you. For these tumor boards please try to identify their schedule: Day (e.g. Friday), frequency (e.g. weekly, bi-weekly, monthly), and time when they start.

**Data Description**

You have **train.csv** and **test.csv** files and folder with corresponding **.html** files.

Files:

* **train.csv** contains next columns: **url**, **doc\_id** and **label**
* **test.csv** contains next columns: **url** and **doc\_id**
* **htmls** contains files with names **{doc\_id}.html**
* **keyword2tumor\_type.csv** contains useful keywords for types of tumorboards

Description of tumor board labels:

* 1 (no evidence): tumor boards are not mentioned on the page
* 2 (medium confidence): tumor boards are mentioned, but the page is not completely dedicated to tumor board description
* 3 (high confidence): page is completely dedicated to the description of tumor board types and dates

You are asked to prepare a model using htmls, referred to in **train.csv**, and make predictions for htmls from **test.csv**

**Practicalities**

You should prepare a Jupyter Notebook with the code that you used for making the predictions and the following documentation:

* How did you decide to handle this amount of data?
* How did you decide to do feature engineering?
* How did you decide which models to try (if you decide to train any models)?
* How did you perform validation of your model?
* What metrics did you measure?
* How do you expect your model to perform on test data (in terms of your metrics)?
* How fast will your algorithm performs and how could you improve its performance if you would have more time?
* How do you think you would be able to improve your algorithm if you would have more data?
* What potential issues do you see with your algorithm?

**Tips**

* to extract clean text from the page you can use BeautifulSoup module like this

from bs import BeautifulSoup

content = read\_html()

soup = BeautifulSoup(content)

clean\_text = soup.get\_text(' ')

If you decide that you don't need, for example, tags **<p>** in your document, you can do this:

from bs import BeautifulSoup

content = read\_html()

soup = BeautifulSoup(content)

for tag in soup.find\_all('p'):

tag.decompose()