**URL Classification**

**Abstract**

This document is for reporting my proposal to the classifying the dataset for “raheem” task. I used scikit-learn package for Decision Trees classifier.

(Ps. worth noting that i received the mail lately and could not have that much privilege of the 72 hours because i’m currently working full time, so it i tried as fast as i can to accomplish this task the easiest way)

**Approach**

After testing various stuff. I found out that the following pipeline is the best (due to deadline):

1- Filtering the data using http request/response protocols:

This method is to guarantee that i have a working URL, so using “get” in requests python library, i filtered those links giving connection error (id status\_code not equal to 200 is considered a corrupted link).

2- after filtering the dataset i splitted it to get the domain name like: [www.{extract\_this}.com](about:blank) to see how likely this website has been used (it’s a counter logic, calculate the frequency of each link, the threshold the counter output to be 5, that is, only the minimum number of sites to be considered have a frequency of 5).

3- the new filtered\_dataset is then scrapped using “URLLIB” and “BeautifulSoap”. The data was so basic so i looked only at the title of the html tags if it contains words from my bag of words:

(mybag = ['raheem', 'rahim', 'ra7em', 'ra7eem', 'ra7im', 'رحيم', 'r7eem', 'rahiem','ra7iem', 'ر7يم'])

Then i assigned them to a new dataset with new architecture(Final\_filtered\_df.csv)

**problems**

1- most of the links contain numbers and blacklist\_characters which is not useful to get infos from so scraping, i assume, is the best option i had, So, i barely used any machine learning algorithm for it because i could not find any feature to be used so at least i can use the famous algorithms like n-grams, all-grams and LDA….etc .

2- some URLs return errors (status\_code not 200 while handshaking with server ) due to different connection types, so if you gotta run the code results will be different every time because number of filtered dataset is not fixed.

3- “Cant Scrape” error is due to domain. The video page redirects me to the home page so when the scaping gets back with the headers it won't be there so it returns this error(actually it’s working it’s not the same as its usual presence, so different attributes return error). That could be handled easily in the API to redirect by number of trials.